UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Programa de Doctorado en Tecnologías de la Información y la Comunicación

Estrategias de pre y postprocesado en deep learning para problemas multiclase en el ámbito de la seguridad y la biodiversidad

Tesis Doctoral

Francisco Pérez Hernández

Granada, enero de 2022

UNIVERSIDAD DE GRANADA

Estrategias de pre y postprocesado en deep learning para problemas multiclase en el ámbito de la seguridad y la biodiversidad

MEMORIA PRESENTADA POR

Francisco Pérez Hernández

PARA OPTAR AL GRADO DE DOCTOR POR LA UNIVERSIDAD DE GRANADA
DENTRO DEL PROGRAMA DE DOCTORADO EN TECNOLOGÍAS DE LA INFORMACIÓN Y LA COMUNICACIÓN

Enero de 2022

DIRECTORES

Siham Tabik y Francisco Herrera Triguero

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Editor: Universidad de Granada. Tesis Doctorales
Autor: Francisco Pérez Hernández
ISBN: 978-84-1117-334-6
URI: http://hdl.handle.net/10481/74944

La memoria titulada “Estrategias de pre y postprocesado en deep learning para problemas multiclase en el ámbito de la seguridad y la biodiversidad” que presenta Francisco Pérez Hernández, ha sido realizada dentro del programa oficial de doctorado en “Tecnologías de la información y la Comunicación” para optar al grado de Doctor bajo la dirección de los doctores Siham Tabik y Francisco Herrera Triguero.

El doctorando Francisco Pérez Hernández y los directores de la tesis Siham Tabik y Francisco Herrera Triguero, garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis y hasta donde nuestro conocimiento alcanza, en la realización de este trabajo, se han respetado los derechos de otros autores a ser citados cuando se han utilizado sus resultados y publicaciones.

Granada, enero de 2022

El Doctorando:

_____

Francisco Pérez Hernández

Los directores de tesis:

_____ _____

Dra. Siham Tabik Dr. Francisco Herrera Triguero

Agradecimientos

Me gustaría comenzar los agradecimientos por mis directores de tesis Francisco Herrera Triguero y Siham Tabik, los cuales depositaron la confianza en mí para hacer que esta tesis se convirtiese en realidad. No se me olvidará la tarde en la que me ofrecieron realizar un doctorado, algo que no habría imaginado nunca y que pensaba que no estaba preparado. Pero el agradecimiento se queda corto ya que me han descubierto un mundo apasionante como es el *deep learning* y sus aplicaciones a visión por computador que es un tema que me apasiona y espero continuar trabajando durante mucho tiempo. Junto a ellos quiero mencionar a mis compañeros, ya que he tenido la enorme suerte de conocer a grandes personas y algunas de ellas se han convertido en grandes amigos. Me gustaría mencionarlos a todos ya que estoy enormemente agradecido, pero quiero hacer mención especial a la persona que empezó conmigo y que ha recorrido junto a mí este camino, Alberto. Mucho ha sido lo que hemos compartido y los ratos que hemos pasado, sin olvidar la sala de servidores.

Por otra parte, se encuentra mi familia, comenzando por mis padres Jose y Basi, y mi hermano Jose, que siempre están apoyando para seguir adelante con más fuerza si cabe y que si he conseguido llegar hasta aquí es por todo lo que me han enseñado. Tengo la mayor suerte del mundo de tenerlos.

Mi pareja Laura se merece una mención especial, ya que es la que me lleva acompañando todo este camino y que sin su apoyo esto no hubiese sido lo mismo, animando en los momentos difíciles y celebrando en los felices. *Thank you very much teacher.*

Tengo la suerte de tener muchos amigos y muy buenos, y ellos también me han acompañado en este camino estando a mi lado y haciendo un hueco para desconectar siempre que lo necesitaba.

Por último, me gustaría dedicar esta tesis, además de a toda mi familia que para mi es lo mejor que tengo y que soy así gracias a todo lo que me ha enseñado cada uno, a mi abuelo Paco, el cual falleció durante este camino, pero que sin duda me sigue acompañando todos los días y se que seguirá ayudándome a cada paso.

Muchas gracias a todos y todas.

Índice general

Resumen

Los modelos de *deep learning* o redes neuronales profundas representan el estado del arte en las tareas principales de visión por computador como clasificación, detección y segmentación en imágenes. De hecho, un tipo concreto de *deep learning* llamado redes neuronales convolucionales superan la precisión humana en la tarea de clasificación de imágenes.

Para afrontar un problema nuevo de un ámbito especifico se usa normalmente la técnica *transfer learning*, que consiste en usar una de las redes más relevantes preentrenada sobre una de las bases de datos de ámbito general más populares, en concreto, ImageNet en clasificación y COCO en detección. Sin embargo, en la mayoría de estos casos, este enfoque produce un numero de falsos positivos y falsos negativos inaceptable.

Abordar problemas nuevos de diferentes ámbitos de la seguridad, como la videovigilancia, o de la biodiversidad, como la seguridad medioambiental, requiere de un modelo robusto que genere un numero mínimo de errores. Diseñar dicho modelo es una tarea compleja por los siguientes motivos:

- El diseño de una nueva base de datos grande y de calidad es costoso, tedioso y manual. En algunos casos, se requiere de conocimiento experto para anotar los datos, lo que complica aun más esta tarea.
- Usar las arquitecturas más potentes del estado del arte en un problema nuevo no es suficiente para obtener modelos que se puedan usar en producción.
- El uso de optimizaciones como *transfer learning* o *fine-tuning* con ImageNet y COCO sigue siendo insuficiente.

Por estos motivos, es necesario el diseño de técnicas específicas de pre y postprocesado *ad hoc* para cada tipo de objetos, imágenes y problemas.

En esta tesis se proponen técnicas novedosas de pre y postprocesado para los modelos *deep learning* de forma que se adapten de forma propia a diferentes problemas de detección con el objetivo de mitigar falsos positivos y falsos negativos en tres aplicaciones del ámbito de la seguridad y la biodiversidad:

1. Detección de objetos pequeños que se manejan de forma similar a un arma con *deep learning* en videovigilancia.
2. Detección de infraestructuras críticas en distintos niveles de zoom con *deep learning* en imágenes de satélite.
3. Detección del cambio en arbustos de alta montaña a través de imágenes de satélite en distintos momentos temporales con *deep learning*.

Introducción

Desde el año 2012, las redes neuronales profundas conocidas como *deep learning*, son el estado del arte en todas las tareas fundamentales de visión por computador, clasificación, detección y segmentación en imágenes. A pesar de ello, la calidad de estos modelos viene supeditada a los datos que se han usado en el entrenamiento de estos, dando grandes resultados en datos de ámbitos similares a los que han sido entrenados, pero obteniendo un alto número de falsos positivos y falsos negativos en otros problemas diferentes.

Las bases de datos de referencia para el preentrenamiento de modelos son de ámbito general en las tareas de visión por computador, ImageNet en clasificación y COCO en detección, pero estos conjuntos de datos presentan el problema de disponer sólo de algunas clases comunes de objetos. Es por ello por lo que para resolver problemas concretos se necesitan crear bases de datos con un objetivo concreto. El diseño de nuevas bases de datos supone una tarea de alta complejidad e incluso se necesita conocimiento experto que valide los datos tomados y realice el etiquetado al no tratarse de objetos comunes.

El uso de técnicas de optimización como *transfer learning* o *fine-tuning* sobre los dataset de referencia, con los que el entrenamiento comienza con unos pesos preentrenados o entrenan solamente unas capas concretas, reducen el tiempo de entrenamiento y se presentan como una herramienta para solucionar multitud de problemas. Aún así, este tipo de optimizaciones no proporcionan resultados de suficiente calidad en entornos específicos.

Como se ha visto, las redes neuronales convolucionales son el estado del arte en problemas de ámbito general tratados con imágenes y vídeos con la pretensión de obtener modelos preparados para ser usados en producción. El problema sucede cuando con las arquitecturas más potentes no se consigue un resultado suficiente para obtener modelos de calidad en nuevos problemas o se obtienen multitud de errores, por lo que requieren una adaptación.

Por estos motivos, existe la necesidad de diseñar técnicas adaptadas de pre y postprocesado para los modelos *deep learning* en cada problema específico y según el tipo de imágenes usadas, de forma que se ajusten a la tarea objetivo del ámbito concreto.

En las siguientes secciones se detalla el tipo de problemas para los que se plantean soluciones en la presente tesis, así como la motivación y los objetivos específicos detrás de la misma y su estructura.

Planteamiento del problema

Según lo que se ha comentado es necesario considerar un conjunto de tareas, que serán descritas con detalle en el Capítulo 1, entre las que destacan la clasificación, detección y segmentación de imágenes. El tipo de dato usado en cada problema también será discutido en este capítulo, viendo diferentes como las imágenes de satélite, imágenes horizontales o vídeos horizontales.

Los modelos del estado del arte que surgen para las tareas de visión por computador en los diferentes tipos de datos junto con sus estrategias de optimización resultan insuficientes a la hora de solventar nuevos problemas con diferentes condiciones ya que no obtienen desempeños similares a los que se consiguen en las competiciones como ImageNet o COCO.

Surge la necesidad de crear técnicas y metodologías específicas tanto de pre como postprocesado de los modelos que se adapten a las condiciones específicas de los datos y la tarea a resolver para cada problema en concreto. A continuación, se introducirán los problemas abordados en esta tesis.

Motivación

Esta tesis aborda tres problemas diferentes en el ámbito de la seguridad y de la biodiversidad, donde todos ellos giran en torno a un denominador común: el diseño de técnicas novedosas de pre y postprocesado para modelos *deep learning* adaptadas a los requerimientos de cada problema:

- El primer problema trata la detección de armas en vídeos de videovigilancia en tiempo real. El mejor modelo actual aún produce un número de falsos positivos inaceptables, especialmente en los objetos que se manejan similarmente a un arma, como una tarjeta o un smartphone.
- El segundo problema aborda la detección de infraestructuras críticas en imágenes de satélite con fuentes gratuitas. La principal limitación en este problema consiste en que las infraestructuras tienen tamaños muy dispares, algunas como las subestaciones eléctricas ocupan una superficie del orden de varios m^2 mientras que infraestructuras como los aeropuertos pueden tener una superficie de hasta decenas de Km^2. Además del problema de intra-variabilidad, la misma infraestructura puede tener aspectos o formas muy diferentes. Para la detección correcta de cada clase de infraestructura se requiere de una resolución concreta.
- El tercero aborda la detección del cambio en arbustos de alta montaña a través de imágenes de satélite en distintos momentos temporales con *deep learning*. Dichos arbustos considerados tienen un patrón espacial muy variable que hace que su correcta segmentación sea compleja.

A partir de este conjunto de planteamientos se definen los objetivos concretos que se abordan en la tesis, enumerados en el siguiente apartado.

Objetivos

Los modelos *deep learning* presentan diversas particularidades en función del problema que se pretende abordar, de los tipos de datos y de cada objeto. El objetivo principal de esta tesis es mejorar la robustez de los modelos *deep learning* a través de la creación de estrategias de pre y postprocesado, en imágenes y vídeos para mitigar falsos positivos y falsos negativos. Las técnicas usadas deberán estar adaptadas a cada escenario concreto, junto con la creación de nuevas bases de datos específicas. Además, el tipo de imágenes o vídeos que se usarán serán distintos, con características particulares.

El objetivo final de esta tesis es:

- Diseño y desarrollo de diferentes estrategias de pre y postprocesado para modelos *deep learning*, que tienen en cuenta las particularidades de los objetos, imágenes y problemas, en los ámbitos de la seguridad y la biodiversidad, a través de imágenes y vídeos.

El objetivo principal se ha dividido en los siguientes objetivos secundarios:

- Mitigar los falsos positivos en escenarios de videovigilancia para reducir falsas alarmas en la detección de armas mediante el diseño de estrategias de postprocesamiento en modelos *deep learning*.
- Detectar automáticamente el nivel de resolución espacial de imágenes de satélite para la detección de infraestructuras críticas a través de estrategias de preprocesamiento en modelos *deep learning*.

- Identificar el porcentaje y el tipo de cambio sufrido en arbustos de alta montaña con el desarrollo de estrategias de postprocesamiento de modelos *deep learning* en imágenes de satélite.

Estos objetivos se han distribuido en la presente memoria de tesis acorde a la estructura descrita en la sección siguiente.

Estructura de la tesis

Además de esta introducción, la presente memoria de tesis consta de cinco capítulos con la siguiente estructura:

- **Capítulo 1:** En el primer capítulo se presentan los aspectos fundamentales para la realización de esta tesis como a) un estudio del estado del arte de los principales modelos de *deep learning*, en especial para las tareas abordadas, detección y segmentación, b) las principales técnicas de pre y postprocesamiento que se pueden aplicar a estos modelos y c) por último, una revisión de los tipos de imágenes y vídeos.

- **Capítulo 2:** En el segundo capítulo se ha abordado el problema de la detección de armas en videovigilancia. El uso de objetos pequeños que se manejan de forma similar a un arma, como smartphone, tarjeta, billete y monedero, produce multitud de falsos positivos y falsos negativos en la detección en entornos donde la seguridad es clave como en bancos o en joyerías. Con el diseño de estrategias de postprocesamiento para modelos *deep learning* se han mitigado los errores cometidos por los modelos en este tipo de escenarios. La metodología desarrollada se basa en dos etapas, una primera encargada de obtener regiones candidatas a contener un objeto a través de un detector de objetos, y una segunda etapa con la que se analizan las regiones propuestas a través del uso de clasificadores binarios y su posterior método de agregación de los resultados. De esta forma se obtiene una salida con conocimiento experto a través de clasificadores especializados. Junto a la metodología se ha creado una base de datos orientada a los objetos que se manejan de forma similar a las armas.

- **Capítulo 3:** En el tercer capítulo se ha afrontado el problema de la detección de infraestructuras críticas en imágenes de satélite donde los objetos presentan multitud de diferencias en función de su resolución espacial junto con grandes cambios tanto inter como intraclase. Para solucionar este problema se ha creado una estrategia de preprocesamiento de modelos *deep learning* que es capaz de detectarla automáticamente. La metodología primero analiza la resolución espacial de la imagen de entrada para después seleccionar un detector experto en función de esta. Cada detector de objetos es experto en un grupo de infraestructuras que están agrupadas en función de sus similitudes. Junto a la metodología se ha diseñado una base de datos de imágenes especializadas para llevar a cabo la detección de infraestructuras críticas como los aeropuertos o las subestaciones eléctricas.

- **Capítulo 4:** En el cuarto capítulo se ha encarado el problema de la identificación del porcentaje y el tipo de cambio sufrido por un arbusto de alta montaña, como es el caso del enebro en las montañas de Sierra Nevada, en imágenes de satélite. Debido al cambio climático y el aumento de las temperaturas el seguimiento temporal de determinadas especies se convierte en una labor de vital importancia para su mantenimiento y la conservación de la biodiversidad. El uso de estrategias de postprocesamiento de modelos *deep learning* permite obtener la diferencia de tamaños entre dos fechas para un mismo arbusto a través del uso de modelos de segmentación. Además de la metodología se ha construido una base de datos especializada en un tipo concreto de arbustos como los enebros.

- **Capítulo 5:** En el quinto capítulo se han resumido las conclusiones obtenidas en cada uno de los distintos trabajos, así como las publicaciones resultantes a raíz de estos. Además, se propone el trabajo futuro que continuará a esta tesis en función de cada uno de los distintos problemas.

Capítulo I

Preliminares

En este primer capítulo se realiza un estudio del estado del arte para distintos modelos de *deep learning* tanto en las tareas de clasificación, detección como segmentación junto con las métricas más extendidas. Además, se elabora un estudio de las estrategias de pre y postprocesado más comunes para estos modelos. Por último, se analizan los diferentes tipos de imágenes y vídeos de los retos abordados con modelos *deep learning*.

1. Modelos *deep learning*

Los modelos *deep learning* se dividen en función de su tarea, siendo clasificación, detección o segmentación las principales:

- **Clasificación:** es la tarea de, dada una imagen, asignarle como salida del modelo una etiqueta de la clase a la imagen completa.
- **Detección:** es la tarea de, dada una imagen, obtener la localización de cada uno de los objetos que aparecen en la imagen y asignarle a cada uno una etiqueta en función de su clase.
- **Segmentación:** es la tarea de, dada una imagen, seleccionar los píxeles que pertenezcan a cada uno de los objetos y asignarles una etiqueta de la clase de cada uno.

Un ejemplo de estas tareas se puede ver en la figura 1.

1.1. Clasificación

Las redes neuronales convolucionales (CNN) son el principal tipo de redes usadas en visión por computar para imágenes. La arquitectura de las redes CNN está inspirada en la percepción visual. Una neurona artificial imita el funcionamiento de una neurona biológica y los núcleos de la CNN representan diferentes receptores que pueden responder a diversas características. Las funciones de activación simulan la función de que sólo las señales eléctricas neuronales que superan un cierto umbral pueden ser transmitidas a la siguiente neurona. Las funciones de pérdida y los optimizadores son elementos utilizados para enseñar a

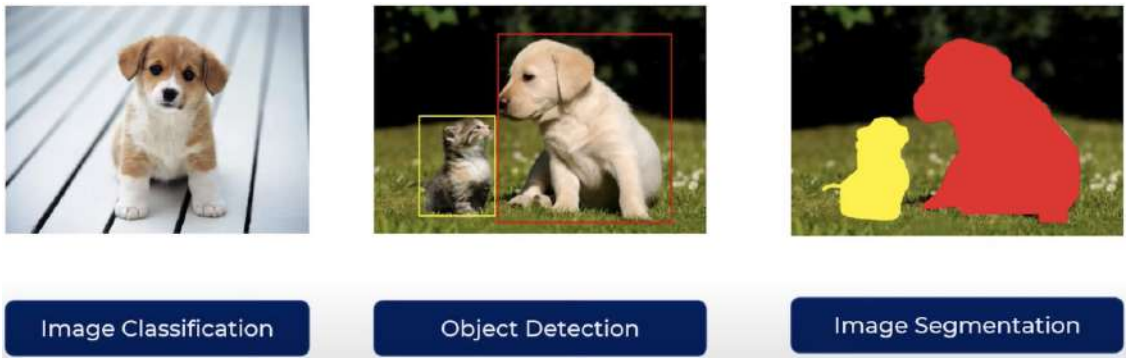


Figura 1: Tareas de *deep learning* como clasificación (izquierda), detección (centro), segmentación (derecha). Obtenida el 03 de diciembre de 2021 de `https://venturebeat.com/2021/05/14/new-deep-learning-model-brings-image-segmentation-to-edge-devices/`

todo el sistema, a todas las neuronas y conexiones del modelo CNN a aprender lo que se espera que extraiga del problema, de los ejemplos.

Desde que se propuso AlexNet en 2012 [KSH12] los investigadores han propuesto una variedad de modelos CNN que son más profundos, más amplios o ligeros. Algunos de los modelos que han surgido son ZFNet en 2013, GoogLeNet, Inception v1 y VGG en 2014, ResNet en 2015, SqueezeNet, Inception v2 y v3 en 2016, Inception v4, SENet, ShuffleNet v1, DenseNet, ResNeXt, Xception y MobileNet v1 en 2017, ShuffleNet v2 y MobileNet v2 en 2018, MobileNet v3 en 2019 y GhostNet en 2020.

A continuación, se describen brevemente las principales redes CNN usadas en esta tesis como ResNet (Microsoft) o Inception (Google, Universidad de Michigan y Universidad de Carolina del Norte).

- **ResNet**

  Las redes neuronales profundas (DNN, *deep neural networks*), pueden extraer características complejas de las imágenes. Sin embargo, con el aumento de las capas, las DNN son propensas a causar problemas como el desvanecimiento y el desajuste de los gradientes. Una de las contribuciones significativas de ResNet [HZRS16] es el bloque residual de dos capas construido por su conexión, como se muestra en la figura 2 (a).

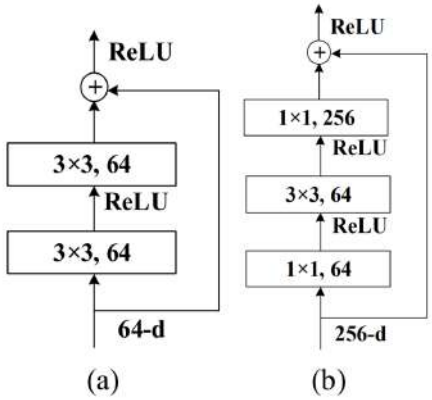


Figura 2: Estructura de los bloques de ResNet. a) Estructura del bloque residual de dos capas. b) Estructura del bloque residual de tres capas. (Obtenida del artículo [LLY+21])

Los modelos ResNet 50, ResNet 101 y ResNet 152, con el número indicando las capas totales del modelo, utilizan bloques residuales de tres capas, como se muestra en la figura 2 (b), en lugar de uno de dos capas. En la figura 3 se muestra la arquitectura de la red VGG19 (parte inferior), red convolucional de 19 capas que ganó la competición ImageNet en 2014 con un error de 7.4 %, y una versión más compacta (34 capas) de ResNet (parte superior), que fue la ganadora de la edición de 2015 con un error de 3.6 %.

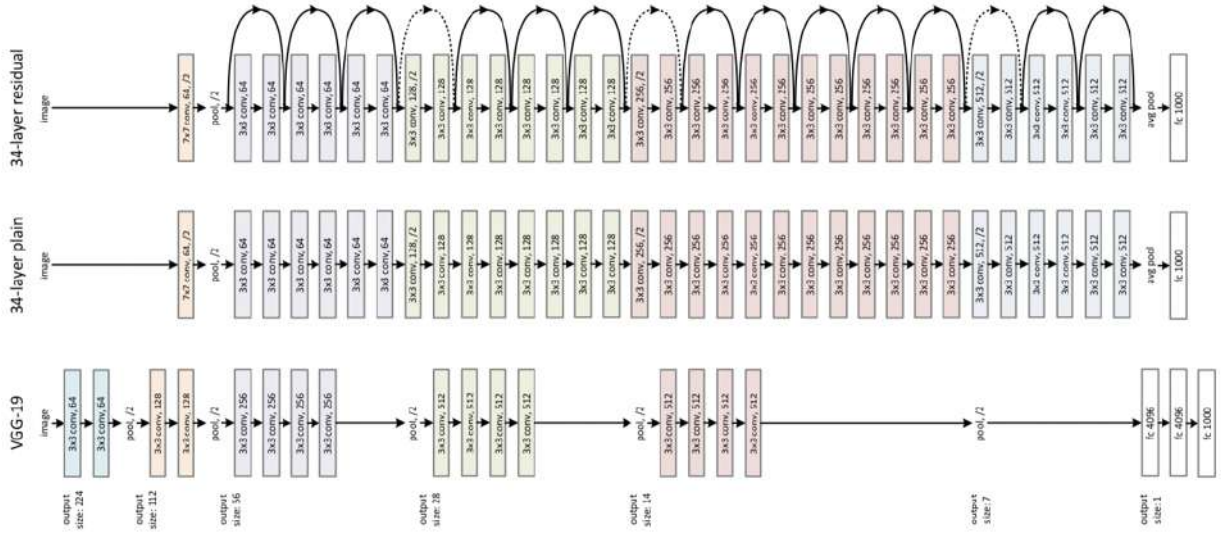
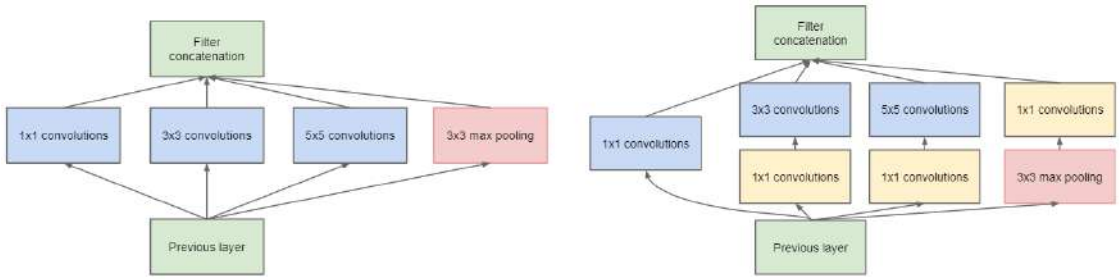


Figura 3: Figura que muestra la arquitectura de la red VGG19 (inferior) junto a la arquitectura de ResNet 34 (superior). (Obtenida del artículo [HZRS16])

- **Inception**

  La red Inception supuso un importante hito en el desarrollo de las CNN. Antes de su publicación, las CNN más populares simplemente se encargaban de hacer las redes más profundas y anchas, con la esperanza de conseguir mejoras de rendimiento. Esto implicaba un gran aumento en el número de los parámetros de la red produciendo problemas de sobreajuste y un incremento exponencial de computación necesaria para el entrenamiento. Por esto, Inception propuso varios cambios para mejorar el rendimiento, tanto en términos de velocidad como en precisión. Debido a su popularidad, varias versiones de esta red han sido desarrolladas en los últimos años. Las más populares son Inception v1, Inception v2, Inception v3, Inception v4 e Inception-ResNet. A continuación, se describen las dos primeras versiones y su combinación con ResNet.

  - Inception v1 [SLJ+15] introduce lo que se conoce como *Inception Modules*, donde se producen convoluciones paralelas con diferentes filtros seguidos de una concatenación. Las convoluciones de $1x1$ son usadas para reducir la dimensión de la entrada con el objetivo de eliminar computaciones costosas. Los filtros $3x3$ y $5x5$ se encargan de capturar características a diferentes escalas. La figura 4 muestra un ejemplo.
  - Inception v2 [SVI+16] utiliza la normalización de lotes para manejar el problema del desplazamiento interno de covariables. La salida de cada capa se normaliza a la distribución normal, lo que puede aumentar la robustez del modelo y entrenar el modelo con una tasa de aprendizaje relativamente grande. Además, Inception v2 muestra que una sola capa convolutiva de $5x5$ pueda ser reemplazada por dos de $3x3$. Sin embargo, el documento original señala que el uso de la



(a) Inception module, naïve version (b) Inception module with dimension reductions

Figura 4: *Inception Modules* v1. a) *Inception Modules* computacionalmente ineficiente. b) *Inception Modules* con reducción de dimensionalidad. (Obtenida del artículo [SLJ+15])

factorización no es efectivo en las primeras capas. Es mejor usarlo en los mapas de características de tamaño medio. Los bancos de filtros deben ser ampliados (más amplios, pero no más profundos) para mejorar las representaciones de altas dimensiones. Por lo tanto, sólo se factoriza la última convolución de $3x3$ de cada rama, mostrado en la figura 5.

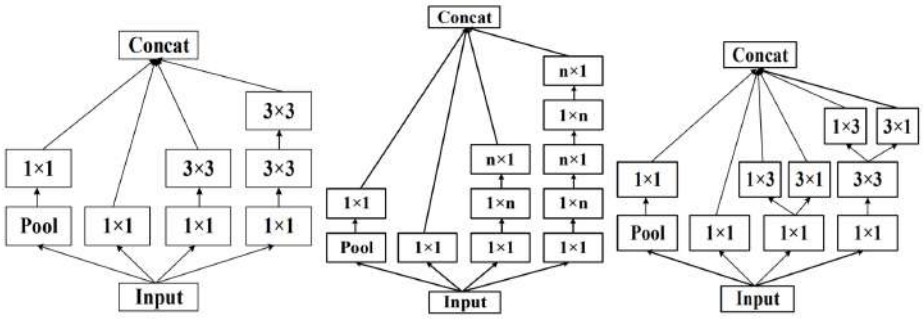


Figura 5: *Inception Modules* v2. Cada convolución $5x5$ es reemplazada por dos convoluciones $3x3$ (izquierda). La convolución nxn es reemplazada por una convolución $1xn$ y una convolución $nx1$ (centro). *Inception Modules* con la última capa convolucional se factorizan (derecha). (Obtenida del artículo [LLY+21])

- Inception-ResNet [SIVA16] es una arquitectura que utiliza *Inception Modules* híbridos, resultado de la combinación de los *Inception Modules* originales y el bloque residual de ResNet (figura 6). Para que los bloques residuales funcionen, la entrada y la salida del módulo después de la convolución deben tener las mismas dimensiones. Por tanto, se usan tantos filtros $1x1$ como la profundidad de la entrada para igualar el tamaño de la entrada y salida. Además, para estabilizar el entrenamiento de la red, se reducen los valores del bloque residual antes de sumarse con la salida de la anterior capa. En general, se escoge un factor de escala entre 0.1 y 0.3 para el escalado.

1.2. Detección

La detección de objetos consiste en localizar los diferentes objetos en una imagen y dar una etiqueta de su clase a cada uno de ellos. Dada una imagen, el modelo la analiza y devuelve:

- Una etiqueta que indica la clase de cada uno de los objetos existentes en la imagen.

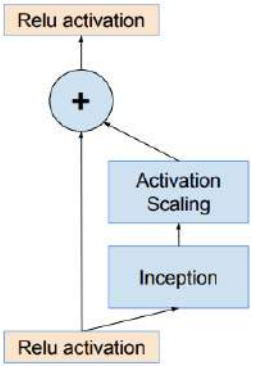


Figura 6: Escalado del bloque residual en Inception-ResNet. (Obtenida del artículo [SIVA16])

- Un rectángulo llamado *bounding box* o cuadro delimitador que indica la localización de cada objeto en la imagen.
- Un número que indica la confianza del modelo en la detección de cada objeto.

Para simplificar la explicación de algunos conceptos básicos, se puede observar como ejemplo el modelo de detección de gatos mostrado en la figura 7. Este modelo realiza la detección del objeto gato, donde tiene que devolver la etiqueta para indicar que la imagen contiene un gato, el *bounding box* que indica dónde se encuentra en la imagen y su confianza.

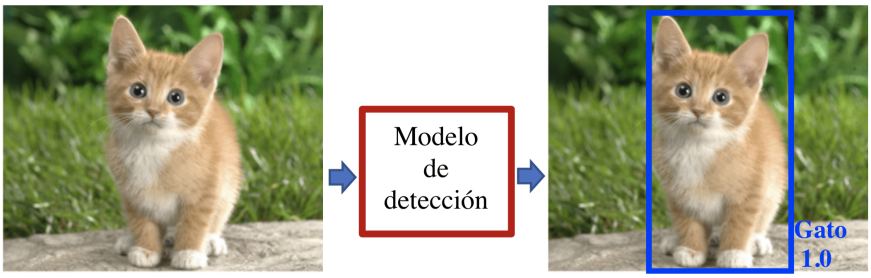


Figura 7: Ilustración gráfica del funcionamiento de un modelo de detección para la clase gato.

Los métodos existentes, basados en aprendizaje profundo y modelos CNN, abordan el problema de detección reformulándolo en un problema de clasificación en dos etapas:

- Entrenar un clasificador con una clase de imágenes asociada al objetivo a detectar.
- Durante el proceso de detección, ejecutar el algoritmo de clasificación CNN sobre varias regiones extraídas de la imagen de entrada. Para la extracción de esas regiones se utilizan diferentes enfoques, *sliding window* o *region proposals*.

A continuación, se describen brevemente los enfoques de extracción de regiones o imágenes con objetos a partir de una imagen original:

- Enfoque *sliding window* [LOB11]: Método exhaustivo que considera un gran número de ventanas candidatas, del orden de 10^4, a partir de la imagen de entrada. Escanea la imagen de entrada, en todos

los lugares y a múltiples escalas, con una ventana y ejecuta el clasificador en cada una de las ventanas. Los trabajos más relevantes en este contexto mejoran el rendimiento de la detección construyendo clasificadores más sofisticados. Las precisiones obtenidas mediante el uso de buenos clasificadores en este enfoque son satisfactorias, pero el proceso de detección puede ser demasiado lento para ser utilizado en tiempo real.

- Enfoque *region proposals*: En lugar de considerar todas las posibles ventanas de la imagen de entrada como candidatas, este enfoque selecciona las regiones candidatas reales utilizando métodos de propuesta de detección. El primer modelo de detección que introdujo las CNN bajo este enfoque fue el de las CNN basadas en regiones (R-CNN) [HBDS15]. Genera alrededor de 2000 cajas delimitadoras potenciales utilizando el método de búsqueda selectiva, deforma las regiones obtenidas en imágenes del mismo tamaño, las alimenta a un potente clasificador basado en CNN para extraer sus características, puntúa las cajas utilizando SVM, ajusta las cajas delimitadoras utilizando un modelo lineal y elimina las detecciones duplicadas mediante una supresión de no máximos.

La figura 8 muestra el resultado final del análisis de una imagen procesada mediante *region proposals*, encontrando en una de ellas un arma, y por tanto el modelo previamente entrenado para detectar armas de fuego daría como salida la detección del objeto.

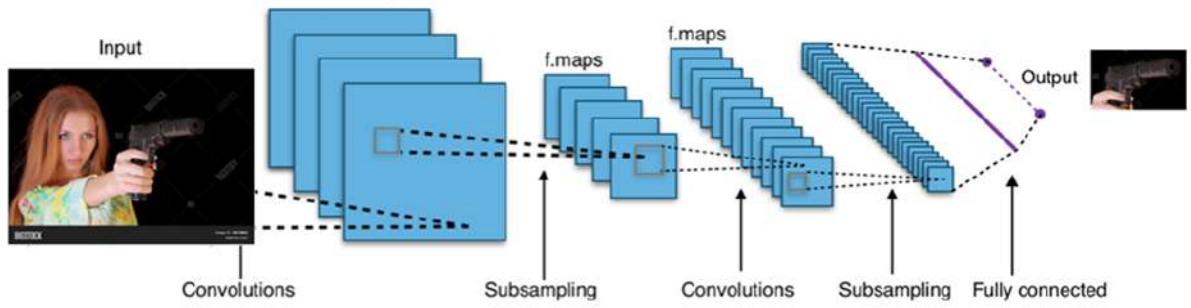


Figura 8: Proceso de la detección de objetos aplicado a armas de fuego. (Obtenida del articulo [OTH18])

Los modelos de detección de objetos han ido evolucionando en los últimos años. El modelo conocido como R-CNN ha evolucionado en modelos más eficientes computacionalmente conocidos como Fast R-CNN y Faster R-CNN[RHGS15a]. Adicionalmente, han surgido nuevos modelos que extienden la idea de regiones como son YOLO (*You Only Look Once*) [RDGF16] y SSD (*Single Shot Multibox Detector*)[LAE$^+$16]:

- **Faster R-CNN:** el modelo deja que la red aprenda las regiones propuestas en lugar de utilizar el algoritmo de búsqueda selectiva, el cuál resulta ser muy lento en la práctica. La imagen se proporciona como entrada a una red CNN de la que se extrae un mapa de características. En lugar de usar aquí la búsqueda selectiva como en anteriores versiones para identificar la región propuesta, se usa una red separada que las predice. Posteriormente, las regiones propuestas por el modelo son modificadas utilizando una capa *pooling*, las cuáles son utilizadas para realizar la clasificación y predecir los valores de desplazamiento de los cuadros delimitadores. Se puede ver una representación del proceso en la figura 9.

- **YOLO:** este modelo, en lugar de seleccionar primero las partes interesantes de una imagen, predice las clases y cajas delimitadores de toda la imagen en una sola pasada del algoritmo. Divide la imagen en celdas, usando una cuadrícula de $19x19$. Cada celda es responsable de predecir 5 cuadros delimitadores, en caso de que haya más de un objeto en esta celda. Por lo tanto, se llega a un número de 1805 cajas

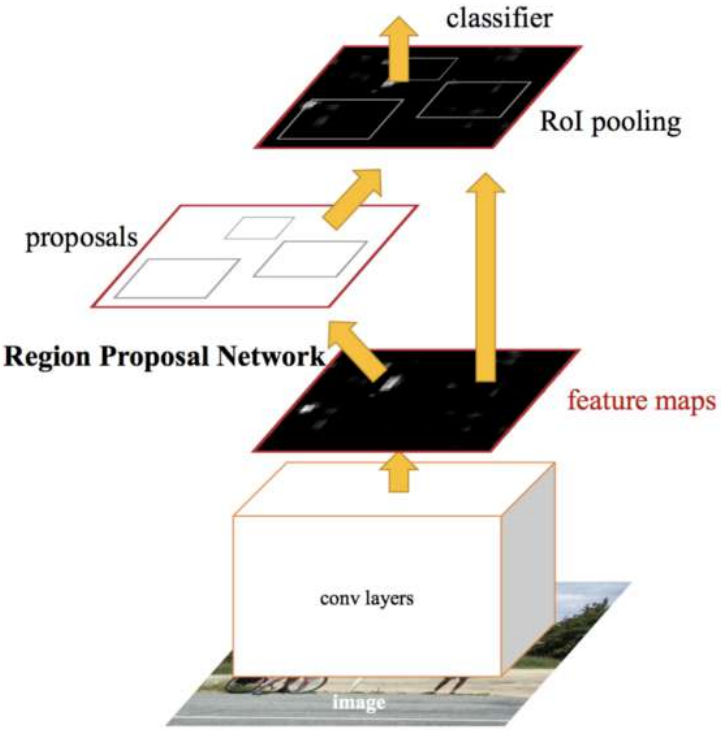


Figura 9: Representación del modelo Faster R-CNN. (Obtenida del articulo [RHGS15a])

delimitadoras para una imagen. La mayoría de estas celdas y cajas delimitadoras no contendrán ningún objeto. Por lo tanto, se predice la confianza, para eliminar las cajas con baja probabilidad y las cajas delimitadoras con el área compartida más alta, en un proceso llamado supresión de no máximos. Se puede ver una representación del modelo en la figura 10.

- **SSD:** es un modelo de detección llamado de un solo disparo. No tiene una red de propuestas de regiones y predice los cuadros delimitadores y las clases directamente de los mapas de características en una sola pasada. Para mejorar la precisión, el modelo SSD introduce:
  - Pequeños filtros convolucionales para predecir las clases de los objetos y compensaciones a las cajas delimitadoras predeterminadas.
  - Filtros separados para las cajas predeterminadas para controlar la diferencia en las relaciones de aspecto.
  - Mapas de características de escala múltiple para la detección de objetos.

    Se puede ver una representación del modelo en la figura 11.

  Además, se realizan más predicciones y se tiene una mejor cobertura en cuanto a ubicación, escala y relación de aspecto. Con las mejoras anteriores, SSD puede reducir la resolución de la imagen de entrada a $300x300$ con un rendimiento mayor para la precisión. Al eliminar la propuesta de región delegada y utilizar imágenes de menor resolución, el modelo puede funcionar en tiempo real obteniendo buena precisión.

Actualmente los modelos de detección que representan el estado del arte son los basados en el enfoque de propuestas de regiones. En esta tesis se usará una de las arquitecturas del estado del arte, Faster

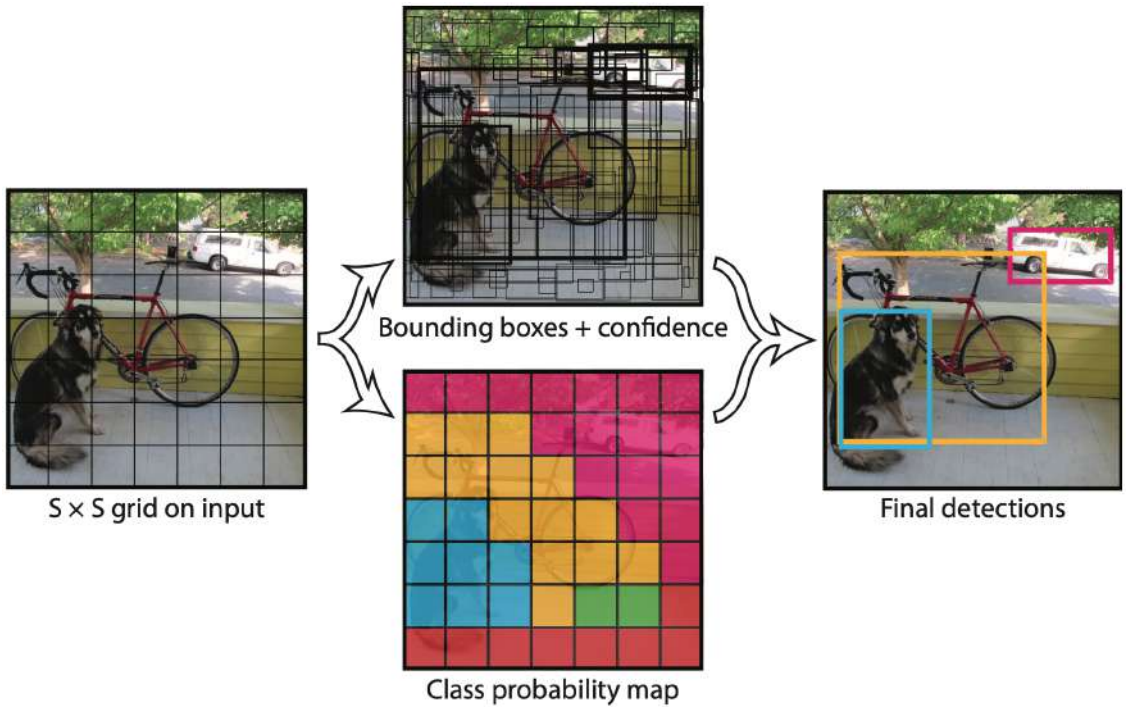


Figura 10: Representación del modelo YOLO. (Obtenida del articulo [RDGF16])

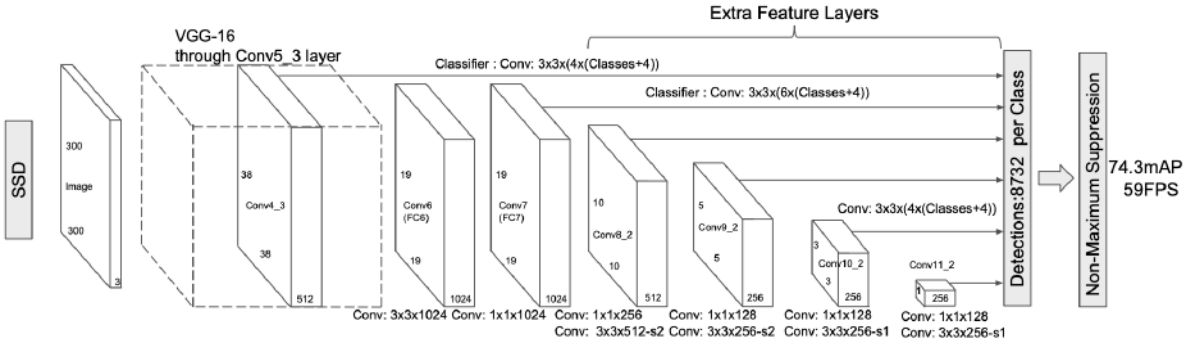


Figura 11: Representación del modelo SSD. (Obtenida del articulo [LAE+16])

R-CNN [RHGS15a], la cuál ofrece un equilibro entre calidad y velocidad necesarias para los problemas que se plantean, además de obtener buenas detecciones sin importar el tamaño del objeto a detectar.

Durante el desarrollo de este trabajo se ha analizado el comportamiento de varios modelos de clasificación que han obtenido grandes resultados en el dataset COCO (*Common Objects in Context*) [LMB+14] y que ofrecen un tiempo de detección por imagen reducido[1]. Estos modelos, que junto con Faster R-CNN forman el modelo de detección completo, son los siguientes:

- **ResNet 101 V1**: Red residual propuesta por Microsoft compuesta por 101 capas [HZRS16]. El modelo formado por Faster R-CNN ResNet 101 V1 ofrece un mAP 0.5-0.95 del 36.6 % en el dataset COCO.
- **ResNet 152 V1**: Red compuesta por 152 capas [HZRS16], que, al ser más profunda, resulta más

[1]Tiempo y mAP de los modelos con la API de TensorFlow: `https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md`

compleja de entrenar. El modelo formado por Faster R-CNN ResNet 152 V1 ofrece un mAP 0.5-0.95 del 37.4 % en el dataset COCO.

- **Inception ResNet V2**: Red compuesta por ResNet e Inception [SLJ+15] dando lugar a grandes resultados en clasificación de imágenes [SIVA16]. El modelo formado por Faster R-CNN Inception ResNet V2 ofrece un mAP 0.5-0.95 del 38.7 % en el dataset COCO.

1.3. Segmentación

Dentro del problema de segmentación nos encontramos con diferentes técnicas como *Semantic Segmentation*, *Instance Segmentation* y *Panoptic Segmentation*:

- ***Semantic Segmentation:*** o segmentación semántica asocia a cada píxel de la imagen una etiqueta de la clase, tratando múltiples objetos de la misma clase como una sola entidad.
- ***Instance Segmentation:*** o segmentación de instancias trata múltiples objetos de la misma clase como instancias individuales distintas.
- ***Panoptic Segmentation:*** o segmentación panóptica combina los conceptos de segmentación semántica y segmentación de instancias asignando dos etiquetas a cada uno de los píxeles de una imagen, una etiqueta semántica y una etiqueta de instancia.

En la figura 12 se puede observar un ejemplo visual de cada uno de los tipos de segmentación.

En esta tesis se ha trabajado con modelos de *Instance Segmentation* por lo que se detallarán algunos de los modelos más importantes del estado del arte.

Las CNN de regiones (R-CNN) y sus variantes han demostrado resultados prometedores en aplicaciones de detección de objetos, en particular el modelo Faster R-CNN [RHGS15b]. Como una extensión de Faster R-CNN, se han creado varios modelos para manejar la tarea de segmentación de instancias. El modelo más famoso, que ya ha superado varios retos de la competición COCO, es Mask R-CNN [HGDG18]. Este modelo realiza simultáneamente la detección de objetos y la generación de una máscara de segmentación para cada instancia. En el mismo marco, *Path Aggregation Network* (PANet) [LQQ+18] fue propuesto utilizando tanto el modelo Mask-RCNN como FPN. Se han creado otros modelos de segmentación de instancias basados en R-CNN, como DeepMask [PCD15], SharpMask [PLCD16], RetinaMask [FSB19], PolarMask [XSS+20], TensorMask [CGHD19], y CenterMask [LP20].

A continuación, se describen brevemente algunos de ellos:

- **Mask R-CNN:** [HGDG17] se desarrolló sobre Faster R-CNN, una CNN basada en regiones. Mientras que Faster R-CNN tiene 2 salidas para cada objeto candidato, una etiqueta de la clase y un desplazamiento del cuadro delimitador, Mask R-CNN es la adición de una tercera rama que genera la máscara del objeto. La salida de la máscara adicional es distinta de las salidas de la clase y la caja, lo que requiere la extracción de un diseño espacial mucho más fino de un objeto. El elemento clave de Mask R-CNN es la alineación píxel a píxel, que es la principal pieza que falta de Fast/Faster R-CNN. Mask R-CNN adopta el mismo procedimiento de dos etapas con una primera etapa idéntica (que es RPN). En la segunda etapa, en paralelo a la predicción del desplazamiento de la clase y caja, Mask R-CNN también genera una máscara binaria para cada RoI. Esto contrasta con los sistemas más recientes, donde la clasificación depende de las predicciones de máscaras. Se puede ver una representación del modelo en la figura 13.

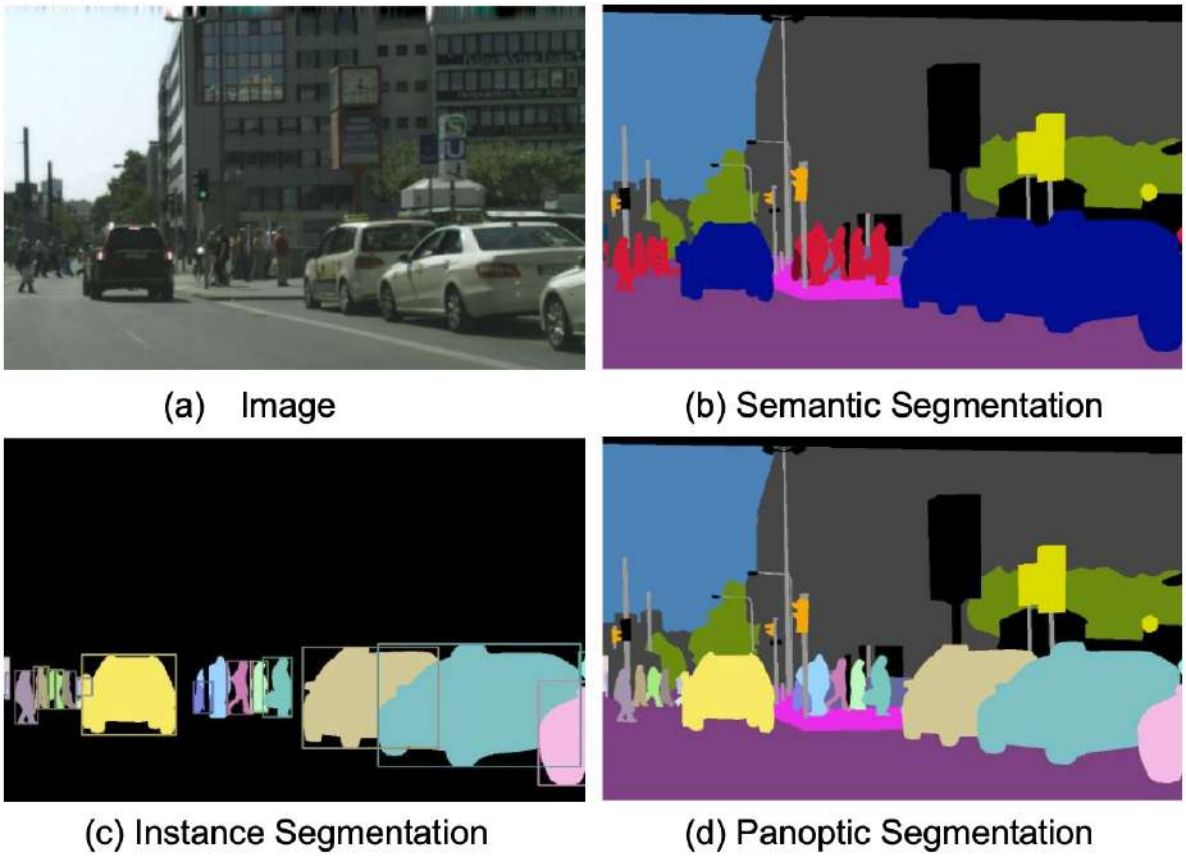


Figura 12: Tipos de segmentación, donde a) es la imagen original, b) *Semantic Segmentation*, c) *Instance Segmentation* y d) *Panoptic Segmentation*. (Obtenida del articulo [CWL+20])

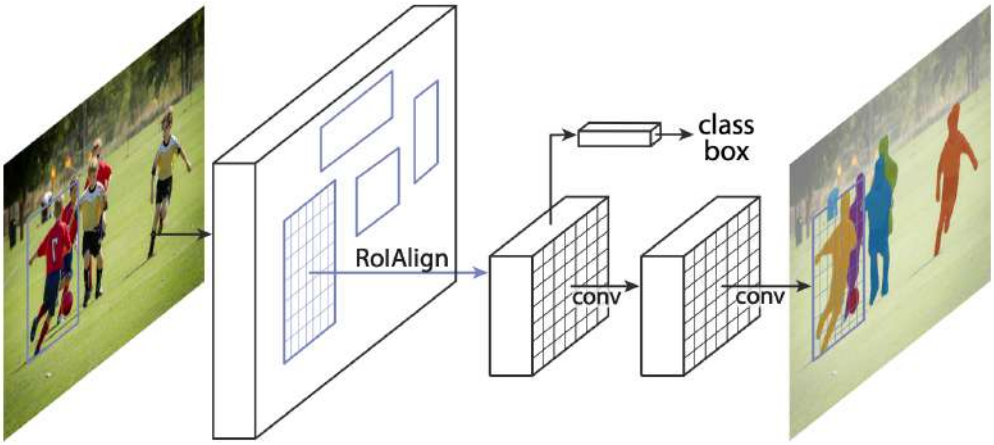


Figura 13: Representación del modelo Mask R-CNN. (Obtenida del articulo [HGDG17])

- **CenterMask:** [LP20] es un método de segmentación de instancias sin anclaje que agrega una nueva rama de máscara guiada por atención espacial (SAG-Mask) al detector de objetos de una etapa (FCOS) sin anclaje, del mismo estilo que Mask R-CNN. Conectado al detector de objetos FCOS, la rama SAG-Mask predice una máscara de segmentación en cada caja detectada con el mapa de atención espacial que ayuda a centrarse en píxeles informativos y suprimir el ruido. Se puede ver una representación del

modelo en la figura 14.

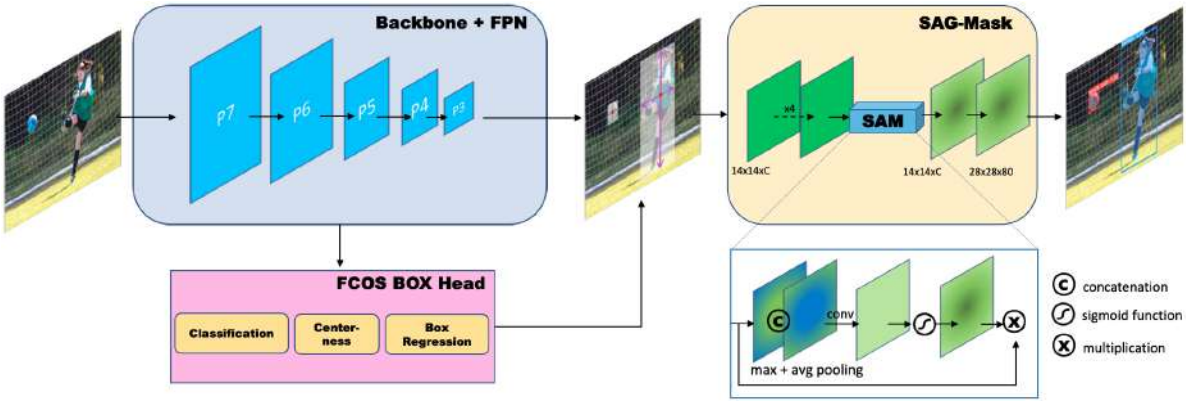


Figura 14: Representación del modelo CenterMask. (Obtenida del articulo [LP20])

- **TensorMask:** [CGHD19] investiga el paradigma de la segmentación de instancias con ventanas deslizantes densas. La principal observación es que esta tarea es fundamentalmente diferente a otras tareas de predicción densa, como la segmentación semántica o la detección de objetos en cajas delimitadoras, ya que la salida en cada ubicación espacial es en sí misma una estructura geométrica con sus propias dimensiones espaciales. Para formalizar esto, se trata la segmentación de instancias densas como una tarea de predicción sobre tensores 4D y se presenta un marco general llamado TensorMask que captura explícitamente esta geometría y permite nuevos operadores sobre tensores 4D. Se demuestra que la visión tensorial conduce a grandes ganancias sobre las líneas de base que ignoran esta estructura, y conducen a resultados comparables a Mask R-CNN. Se puede ver una representación del modelo en la figura 15.

1.4. Métricas de evaluación

Los modelos de clasificación se evalúan por las métricas precisión, *recall*, y F1 [JHP+19]. La precisión (también llamada valor predictivo positivo, es decir, cuántos objetos detectados son verdaderos), el *recall* (también conocido como sensibilidad, es decir, cuántos objetos reales se detectaron), y la medida F1, que evalúa el equilibrio entre la precisión y el *recall*. Donde

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Estas medidas se basan en los Falsos Positivos (FP), Verdaderos Positivos (TP) y Falsos Negativos (FN). Se produce un FN cuando el modelo de detección falla en encontrar el objeto de interés, un TP cuando el modelo detecta correctamente el objeto y un FP cuando el modelo realiza una detección errónea.

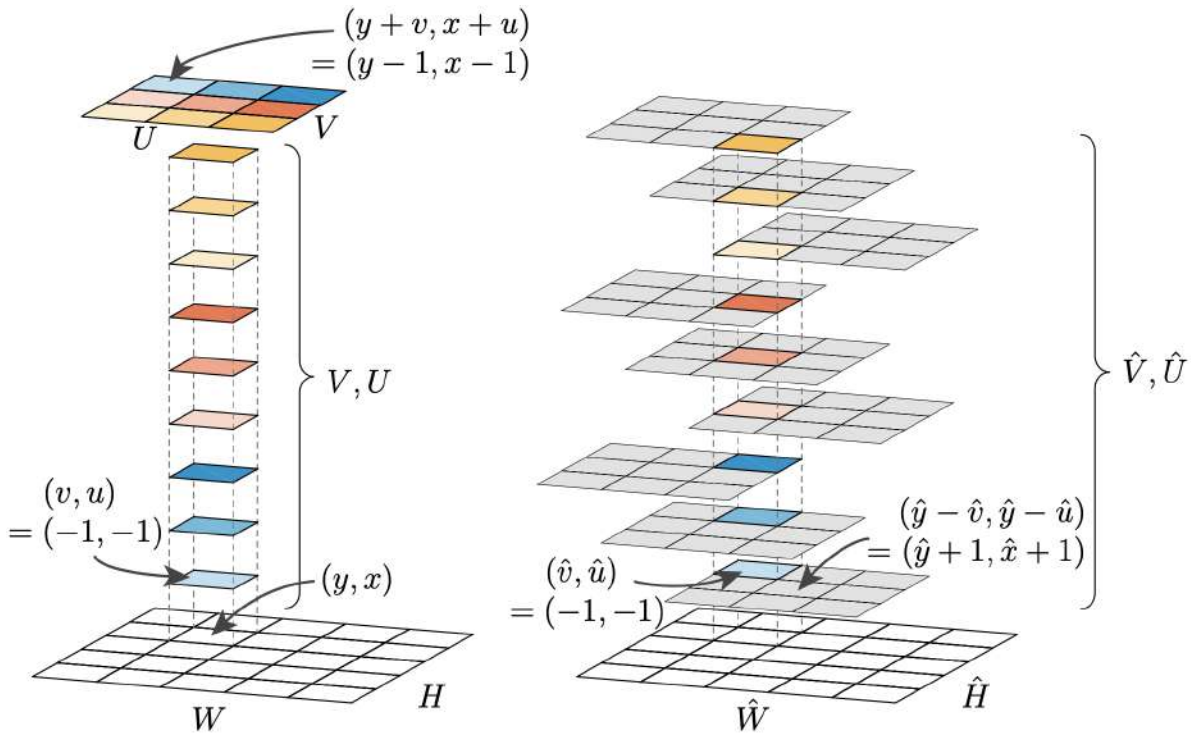


Figura 15: Representación del modelo TensorMask, a la izquierda una representación natural y a la derecha una representación alineada. (Obtenida del articulo [CGHD19])

En los modelos de detección, para validar que se localiza correctamente el objeto en la imagen, se usan métricas diferentes apoyadas en las de clasificación. Estas métricas son el mAP (*mean average precision*) y el mAR (*mean average recall*):

$$mAP = \frac{\sum_{i=1}^{K} AP_i}{K} \qquad AP_i = \frac{1}{10} \sum_{r\epsilon[0,5,\ldots,0,95]} \int_0^1 p(r)dr$$

$$mAR = \frac{\sum_{i=1}^{K} AR_i}{K} \qquad AR_i = 2 \int_{0,5}^{1} recall(o)do$$

donde, K es el número de categorías, p representa la precisión y r *recall* define el área bajo la curva de interpolación $precision - recall$ para cada clase i, mientras que o es la intersección sobre la unión (IoU) y $recall(o)$ corresponde al *recall* bajo la curva $recall - IoU$ para cada clase i.

Concretamente en los experimentos realizados se hará uso de las métricas descritas, y para mAP y mAR se usará un mAP del 0.5 de IoU para analizar cada clase en particular y se usará un mAP y mAR del 0.5-0.95, es decir, una IoU que va del 0.5 al 0.95 pasando de 0.05 en 0.05, para analizar el comportamiento general del modelo. El valor de mAP del 0.5 significaría que el solapamiento que se produce entre un objeto detectado y su verdadera posición se debe corresponder como mínimo en el 50 % para poder ser considerado como válido. En el caso del mAP o mAR del 0.5-0.95 se hace el mismo proceso que para 0.5 pero avanzando de 0.05 en 0.05 hasta llegar a 0.95, siendo el 0.95 una correspondencia del 95 %.

Además de producir valores de mAP y mAR globales se ofrecen resultados por áreas, es decir, según el área que ocupa un objeto. Hay tres niveles de área como *small*, *medium* y *large*, que se determinan según

el número de píxeles que ocupa en la imagen un objeto. Se obtendrá un área *small* cuando el área ocupada por el objeto sea menor que 32^2 píxeles, *large* cuando el área sea mayor que 96^2 y *medium* para los valores intermedios.

Para el problema de segmentación la métrica mAP es la más extendida y se usa como criterio de evaluación calculada por píxel mediante el IoU.

2. Estrategias de pre y postprocesado

Los modelos *deep learning* del estado del arte en tareas de visión por computador obtienen grandes resultados en base a los datos en los que han sido entrenados. Actualmente, los modelos que trabajan con imágenes o vídeos más precisos se basan en CNN profundas [KSH12, ZYCL18]. Las CNN aprenden automáticamente las características distintivas de los objetos a partir de la extracción de características de un gran conjunto de datos etiquetados [YHX+19].

Estos modelos por si solos no son suficientes a la hora de resolver problemas más complejos y que requieren de una base de datos específica. Es por ello por lo que se hace necesario el uso de técnicas de pre y postprocesamiento para los modelos, de forma que se adapten al problema concreto. Algunas opciones, una vez creado el dataset especializado, son el *transfer learning*, *fine-tuning*, *data augmentation*, *ensemble*, etc.

ImageNet Large Scale Visual Recognition Challenge[2] (ILSVRC) [RDS+15] y *Common Objects in Context* (COCO)[3] [LMB+14] son dos de los conjuntos de datos más conocidos en el ámbito de la clasificación y detección de imágenes, dando nombre a dos de las competiciones y benchmarks de referencia. El modelo de detección que ganó la competición ILSVRC en 2017 logró un mAP del 73 % en un conjunto de datos de 527892 imágenes organizadas en 200 clases de objetos [HSS17]. El modelo de detección que ganó el reto COCO en 2017 logró un mAP del 73 % en un conjunto de datos organizado en 80 clases. Los mejores resultados para objetos grandes fueron el mAP del 66 % y el mAR del 82 %, mientras que un mAP de un 34 % y un mAR de 52 %, se obtuvieron en objetos pequeños. En general, los modelos de detección con mejores resultados y más robustos combinan una meta-arquitectura, como Faster-RCNN o R-FCN [RHGS15a, DRF16], con una de las arquitecturas de clasificación del estado del arte basadas en ResNet, VGG o Inception [HZRS16, SZ14, SLJ+15].

El entrenamiento de los modelos *deep learning* requiere de una cantidad amplia de datos para el correcto aprendizaje. Para ello, es posible obtener un conjunto de datos del estado del arte, pero estas bases de datos normalmente presentan clases de objetos comunes. Para resolver determinados problemas se presenta la problemática de la recolección de nuevos datos, donde en muchas de las situaciones se necesita un experto que valide que los datos son correctos. Esta adquisición lleva implicada la creación del etiquetado de los datos, aspecto que se lleva a cabo de forma manual en la mayoría de las situaciones. Para ello, hay diferentes herramientas como pueden ser *LabelImg*[4] para imágenes de detección (figura 16) o *LabelMe*[5] para imágenes de segmentación (figura 17). Un ejemplo de cada uno de estos programas se observa en las figuras 16 y 17, donde el etiquetado se tiene que realizar de forma manual.

La cantidad de datos es un aspecto fundamental para que los modelos obtengan resultados de calidad. Para incrementar la cantidad de estos existen técnicas de preprocesamiento como *Data Augmentation* (DA) [SK19, TPHPH17], que consisten principalmente en aplicar variaciones a los datos para ampliar el número

[2]ImageNet: `https://www.image-net.org` recuperada el 10 de diciembre de 2021.
[3]COCO: `http://cocodataset.org/#detection-leaderboard` recuperada el 10 de diciembre de 2021.
[4]LabelImg: `https://github.com/tzutalin/labelImg` recuperada el 02 de diciembre de 2021.
[5]LabelMe: `https://github.com/wkentaro/labelme` recuperada el 02 de diciembre de 2021.

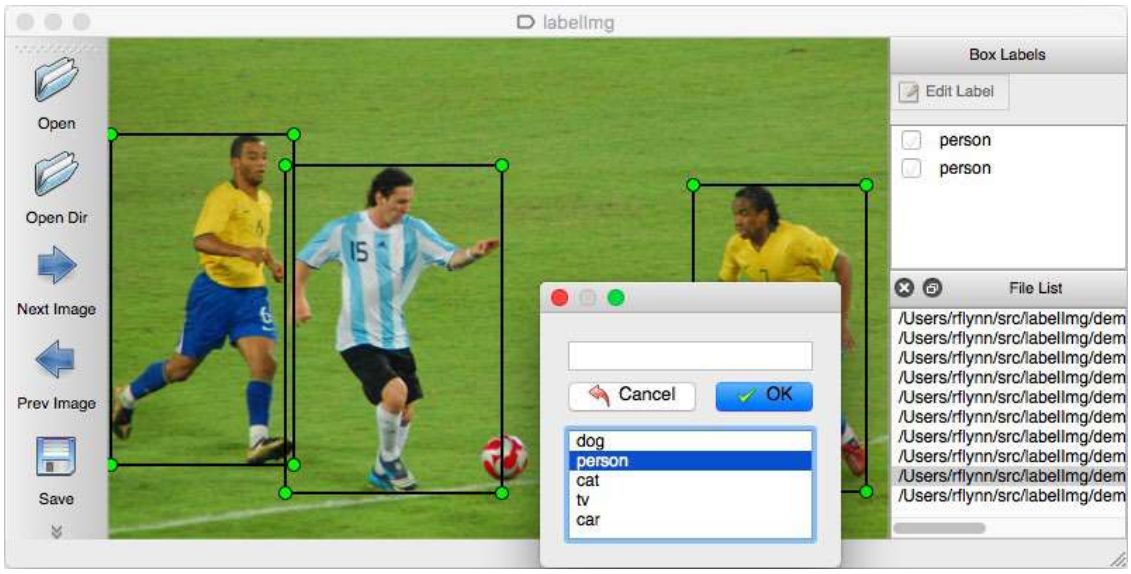


Figura 16: Herramientas para el etiquetado de imágenes, *LabelImg* para problemas de detección.

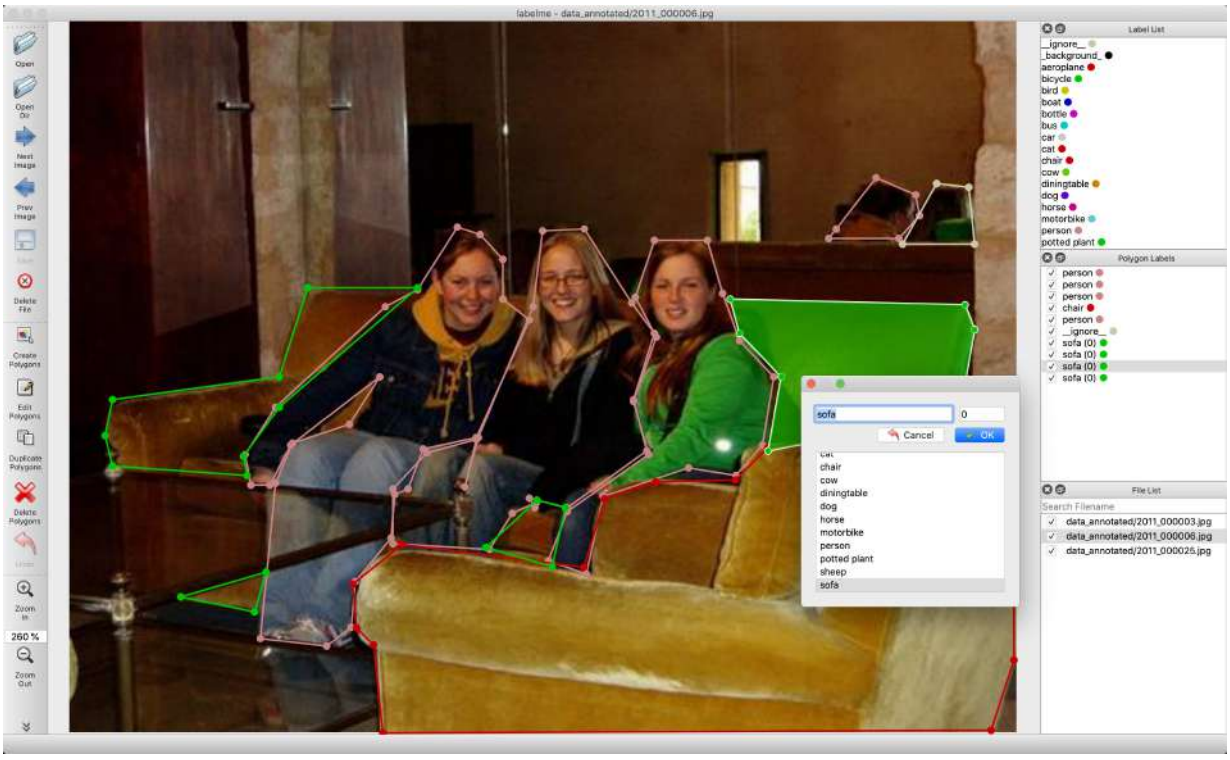


Figura 17: Herramientas para el etiquetado de imágenes, *LabelMe* para problemas de segmentación.

de ejemplos con los que el modelo entrenará. De esta forma, se consigue que el modelo sea más generalizable, pudiendo adaptarse a situaciones que antes no conocía y que aspectos exógenos no interfieran al rendimiento.

La figura 18 muestra algunas de las técnicas de DA que se aplican sobre imágenes. Se observa como en la imagen se realizan rotaciones, recortes, giros, etc. Son muchas las APIs sobre DA que se han creado

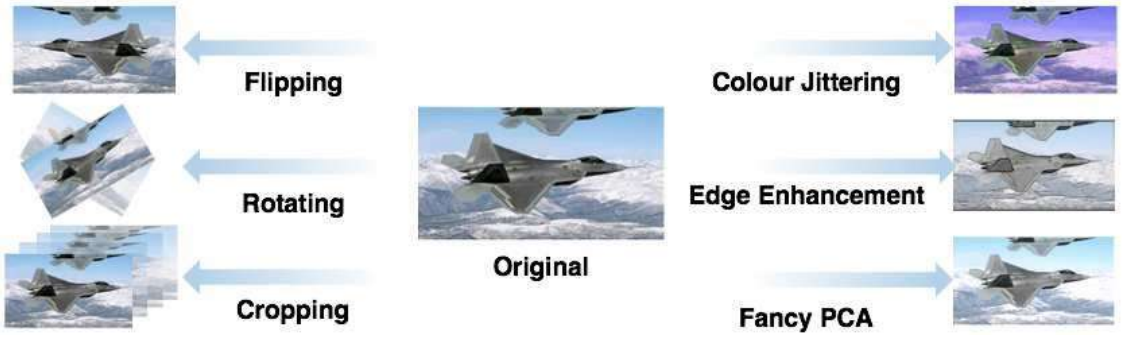


Figura 18: Ejemplo de distintas técnicas de *Data Augmentation*. (Obtenida del articulo [TN18])

para realizar estas técnicas sobre los datos como es el caso de *Albumentations*[6], pero no todas las técnicas son aplicables a todas las imágenes. Para ello es necesario conocer las características de los datos y de cada problema en concreto. Un ejemplo podría ser en la detección de vehículos, donde usar la técnica de DA para girar verticalmente la imagen podría no ser aconsejable, ya que en la práctica no se realizarían detecciones de vehículos volcados.

Además de la cantidad de datos, para obtener modelos con buenos rendimientos, es fundamental tener datos de calidad. Para cada problema se necesitan estudiar las condiciones en las que el modelo deberá trabajar. En el caso de la detección de armas, entrenar el modelo con imágenes en las que el arma aparezca borrosa podría suponer que, en situaciones reales, cuando aparezca un arma difuminada, el sistema dé una señal de alarma a la policía de forma errónea. Para ello el preprocesado de la base de datos es necesario eliminando instancias que causen confusión.

El entrenamiento de un modelo *deep learning*, y en especial una CNN, requiere del ajuste de millones de parámetros. Para que, a la hora de entrenar un nuevo modelo, estos parámetros no se tengan que ajustar desde el inicio, existe la técnica *transfer learning* [PY10]. De esta forma, se usan modelos preentrenados sobre una base de datos de millones de instancias como ImageNet como punto de partida para inicializar los pesos de la red, aprovechando así conocimiento previamente aprendido. Con la técnica *fine-tuning*, el nuevo modelo solo tiene que entrenar las últimas capas y alcanzar así resultados favorables en menos tiempo usando menos cómputo. Estas técnicas obtienen buenos resultados en casos en los que el dataset preentrenado está basado en problemas de características similares.

Algunas de las técnicas de postprocesado para *deep learning* mas conocidas son los *ensemble* [DP14], técnica que consiste en incrustar uno o varios modelos. Esto consigue reunir a varios modelos especializados en diferentes aspectos para obtener salidas más expertas que realizarlo solamente con modelos genéricos. Es el caso del uso de las técnicas de binarización como *One-Versus-All* (OVA) y *One-Versus-One* (OVO), consistentes en el entrenamiento de modelos binarios expertos para una posterior agregación de las salidas de cada uno de los modelos. OVO se compone de modelos binarios en los que cada clase se enfrenta a otra. OVA realiza tantos modelos binarios como clases tenga el problema, enfrentando en cada modelo una clase frente al resto de objetos entrenados como una case conjunta nueva.

[6]Albumentations: `https://albumentations.ai` recuperada el 02 de diciembre de 2021.

3. Tipos de imágenes y vídeos

Las imágenes son el tipo de dato que usan las CNN para extraer características de estas y entrenar los modelos, las cuales pueden ser de diferentes tipos y características en función del problema que se quiera resolver. Además de imágenes, el uso de vídeos sirve tanto para extraer frames para el entrenamiento de los modelos como para realizar inferencia de los modelos y comprobar que estos son capaces de funcionar en tiempo real.

Los principales tipos de datos que se han usado en esta tesis han sido:

- **Imágenes horizontales:** Tipo de imagen que trata de recoger información del mundo real como podría ser vista por una persona. Este tipo de dato es comúnmente usado cuando se quiere realizar la detección de objetos comunes o que pueden aparecer en el día a día. Son muchas las bases de datos del estado del arte con este tipo de imágenes, como ImageNet o COCO, pero cuando una clase no está incluida en estas bases de datos es necesario llevar a cabo su creación. La recolección de los datos se puede realizar tanto con la descarga de imágenes de Internet, como a través de la creación de contenido propio con el uso de cámaras, desde cámaras móviles hasta profesionales. Algunos ejemplos de estas fotografías se pueden ver en la figura 19.

Figura 19: Ejemplos de imágenes horizontales usadas para la detección de armas.

- **Vídeos horizontales:** Los vídeos horizontales son similares a las imágenes horizontales, pero están compuestos de una consecución de frames donde se realizan acciones. El uso de este tipo de datos hace posible tener a disposición multitud de poses y perspectivas de los diferentes objetos, así como un amplio rango de posibilidades de su uso que con imágenes sería más difícil conseguir. Algunos frames de estos vídeos se pueden ver en la figura 20.

- **Imágenes de satélite:** Las imágenes de satélite son imágenes verticales en las que se tiene una perspectiva superior del mundo. Este tipo de imágenes se toman normalmente con un satélite o con aviones, aunque también podrían ser capturadas por drones. El objetivo de este tipo de imágenes es el análisis de la superficie terrestre y de los cambios que se produzcan en el mismo. Algunas de las bases de datos más conocidas son DOTA [XBD+18] y DIOR [LWC+20], las cuales contienen clases comunes como piscina, rotonda, campo de fútbol, etc. Para la obtención de nuevas imágenes es posible acceder a fuentes públicas como son Google Maps[7] o Bing Maps[8], junto a otras fuentes privadas como

[7]Google Maps API: `https://cloud.google.com/maps-platform`
[8]Bing Maps: `https://www.bing.com/maps`

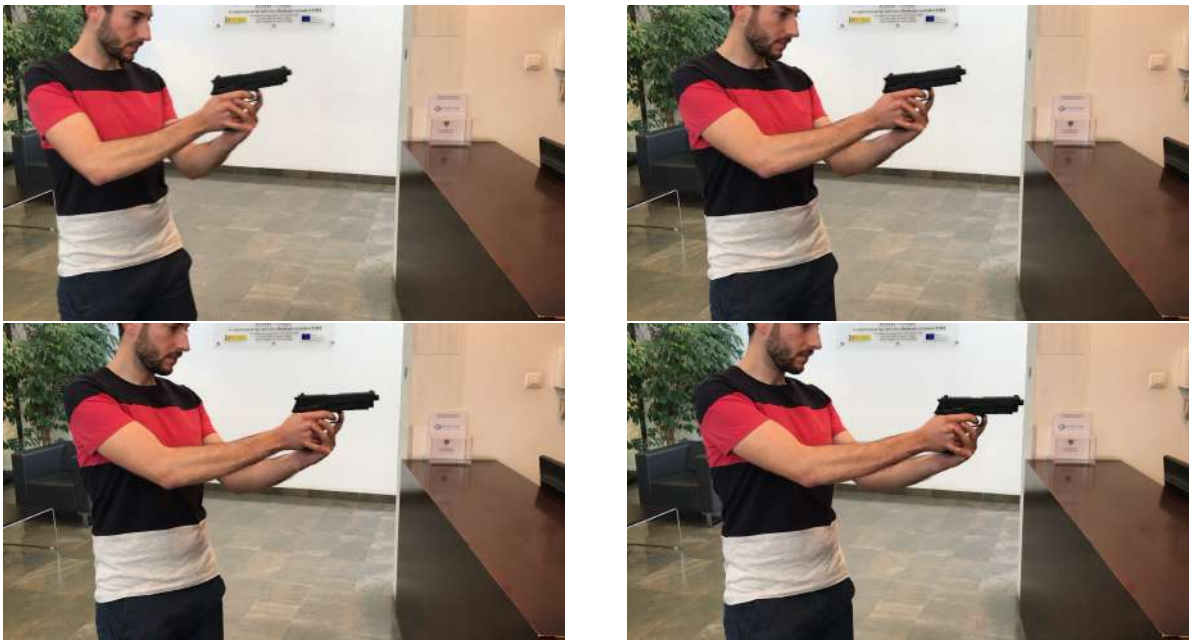


Figura 20: Ejemplos de frames de vídeos horizontales usados para la detección de armas.

PlanetLabs[9] o Airbus Intelligence[10]. Estas imágenes se pueden obtener a distintas distancias o niveles de zoom, cuanto mayor es este valor, de mayor tamaño se pueden ver los objetos y más detalles puede tener cada uno. En la figura 21 se observan algunos ejemplos de dos zonas a distintos niveles de zoom y de una imagen de arbustos de alta montaña.

Figura 21: Ejemplos de los niveles de zoom 19 (izquierda) y 20 (centro) en una zona determinada, y de una imagen de arbustos de alta montaña (derecha).

[9]PlanetLabs: `https://www.planet.com`
[10]Airbus Intelligence: `https://www.intelligence-airbusds.com`

Capítulo II

Metodología ODeBiC para la detección de armas con *deep learning* en vídeos

En este capítulo se propone el desarrollo de una metodología basada en una técnica de postprocesamiento de modelos *deep learning* para el problema de la detección de armas y objetos pequeños que se manejan similarmente. Algunas de las situaciones en las que este tipo de sistemas va a trabajar son colas de bancos o en establecimientos como joyerías o gasolineras en las que la detección temprana de armas puede evitar delitos e incluso muertes.

La detección de armas en vídeos de seguridad es una tarea compleja ya que un sistema encargado de lanzar señales de alarma en este tipo de situaciones debe dar resultados de calidad y sin cometer errores con otros objetos.

En este tipo de escenarios, el uso de otros objetos es frecuente por lo que los modelos pueden obtener multitud de falsos positivos y falsos negativos en estos objetos. Algunas de las clases abordadas son los smartphone, billete, monedero y tarjeta, objetos pequeños que se manejan similarmente a un arma. Para ello, es necesaria la creación de una base de datos especializada en estos objetos, haciendo uso de las bases de datos de pistola y cuchillo presentes en la literatura.

Para abordar el problema de detección que se presenta se ha diseñado la metodología ODeBiC con la que se hace uso de una técnica de postprocesamiento de modelos *deep learning* para mejorar al modelo anterior. Esta metodología está compuesta de dos fases donde la primera de ellas es la encarga de obtener regiones candidatas de la imagen de entrada. Estas regiones serán seleccionadas por un detector el cual ha sido entrenado sobre todas las clases objetivo por lo que las regiones tendrán una alta probabilidad de ser una de las clases que se quieren obtener. Cada una de estas regiones es enviada a cada uno de los modelos de clasificación binarios con el que se obtienen tantas salidas como clasificadores haya. La salida del objeto final se obtiene con el uso de una técnica de agregación.

El uso de la metodología ODeBiC para el problema de la detección de armas y objetos que se manejan de forma similar reduce el número de falsos positivos hasta en un 56.50 %, mejorado considerablemente al detector base.

1. Antecedentes y trabajos relacionados

En esta sección se realiza la revisión del estado del arte para el problema de la detección de armas. Primero se analizan los antecedentes en la subsección 1.1. La subsección 1.2 proporciona un resumen de los trabajos relacionados que utilizan estrategias de binarización, el estado del arte en la detección de objetos en imágenes y los estudios que abordan la detección de armas en vídeos. En la subsección 1.3 y 1.4 se presenta un breve resumen de los métodos de binarización *One-Versus-All* (OVA) y *One-Versus-One* (OVO) respectivamente.

1.1. Antecedentes

Muchos problemas del mundo real requieren la detección de múltiples objetos en imágenes o vídeos [PPR18]. La construcción de detectores útiles para este tipo de problemas puede resolverse utilizando modelos modernos de *deep learning*, especialmente cuando las clases objetivo son diferentes, es decir, de distinto tamaño, color, forma y textura. Sin embargo, esta tarea se complica cuando las clases objetivo son pequeñas (representadas por un número reducido de píxeles, tamaño, forma, color y textura similares) y se manejan de forma similar.

La capacidad de distinguir entre objetos pequeños manipulados con la mano es esencial en varios campos, especialmente en la videovigilancia, donde la detección correcta es extremadamente importante. Un caso importante para la prevención de la violencia es la detección de armas en lugares, como bancos o joyerías, donde la gente suele manipular objetos cotidianos que pueden ser confundidos con una pistola o un cuchillo ya que se manejan de forma similar y que podrían ser smartphone, billete, monedero y tarjeta.

Es posible pensar que hoy en día el número de homicidios ha decrecido debido a los sistemas de seguridad y los agentes de vigilancia que se han implantado recientemente, pero el FBI ha publicado que el 77 % de los homicidios en 2020 se cometieron con armas de fuego, el porcentaje más alto jamás registrado[1]. Abordar este problema es de vital importancia ya que muchos de los crímenes que se cometen podrían ser prevenidos con la detección temprana de estas armas. Este sistema sería aplicable a centros comerciales o zonas con gran cantidad de cámaras de seguridad donde un guarda de seguridad no podría estar atento a todas las cámaras a la vez y la seguridad de los ciudadanos podría verse comprometida.

Este es un problema en el que con el avance del *deep learning* y las distintas técnicas de visión por computador se podría prevenir y dar solución con el uso automático de la tecnología, pero para ello un modelo de *deep learning* del estado del arte no obtiene una precisión suficientemente alta y sin errores. Por ello, en este trabajo se propone una estrategia de postprocesamiento de los modelos para solventar los problemas que se presentan.

Por otro lado, las técnicas de binarización como OVA [CB91, AMMR95] y OVO [KPD90, PCST00, Abe03] convierten un problema multiclase en varios modelos binarios expertos y calculan la clase final mediante un método de agregación. Estas técnicas se utilizan a menudo para reducir la inestabilidad en problemas desequilibrados [ZBX+19, FGG+18] y presentan un gran potencial para el problema de detección de objetos similares.

Este trabajo propone una metodología precisa y robusta, *Object Detection with Binary Classifiers based on deep learning* (Metodología ODeBiC), para la detección de objetos pequeños manipulados de forma similar a un arma aplicada a vídeos de vigilancia.

[1]Noticia publicada en la BBC: `https://www.bbc.com/mundo/noticias-internacional-58714578` recuperada el 29 de noviembre de 2021.

El primer modelo para la detección de armas en vídeos fue propuesto por Olmos et al. [OTH18], donde los autores formularon el problema en un problema de dos clases (pistola vs fondo), construyeron una base de datos de entrenamiento utilizando imágenes de Internet y utilizaron Faster-RCNN basado en VGG16 como modelo de detección. Un ejemplo de este sistema se puede ver en la figura 22.

Figura 22: Sistema de detección de armas de fuego. (Obtenida del articulo [OTH18])

En general, este modelo consigue buenos resultados, pero confunde la pistola con objetos que pueden manejarse de forma similar, por ejemplo, cuchillo, smartphone, billete, monedero y tarjeta. La figura 23 muestra algunos de estos falsos positivos. Estos resultados muestran como el modelo considera que la forma de manipular las pistolas es una característica clave de la clase pistola, lo que supone un problema desde el punto de vista de la videovigilancia. Se aborda este caso de estudio con la metodología ODeBiC, con el objetivo de mejorar la detección de los objetos pequeños manipulados de forma similar.

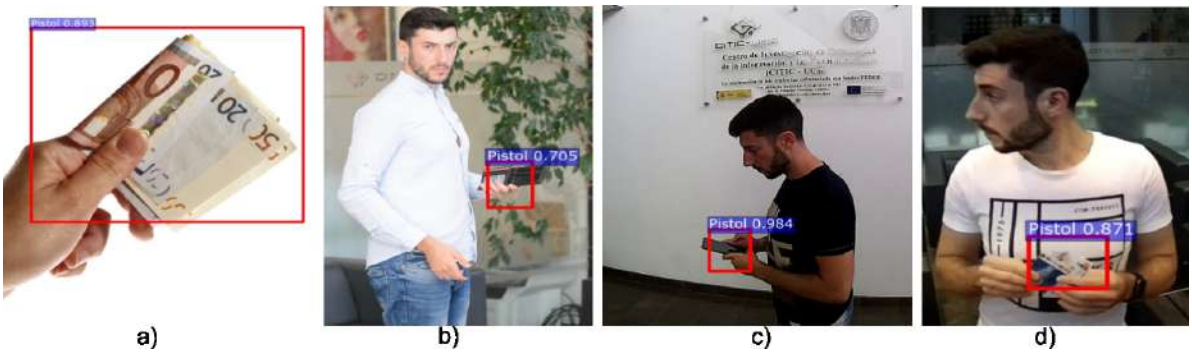


Figura 23: Falsos positivos cometidos por el modelo propuesto en [OTH18], donde los objetos son a) billete, b) monedero, c) smartphone y d) tarjeta.

Las principales aportaciones de este trabajo son:

- Se propone y evalúa una metodología de dos niveles denominada ODeBiC, basada en el uso de *deep learning*, para mejorar la detección de objetos pequeños que pueden ser tratados de forma similar a un arma. El primer nivel utiliza un detector para seleccionar de cada fotograma de entrada las regiones candidatas con una confianza específica sobre la presencia de cada objeto. A continuación, el segundo nivel analiza estas propuestas mediante una técnica de binarización para identificar los objetos con

mayor precisión. La metodología ODeBiC mantiene una buena precisión también para la detección de objetos grandes.

- Se analiza el potencial de las técnicas de binarización como, OVA y OVO, para mejorar la detección de objetos pequeños, manipulados con la mano, que pueden ser confundidos con un arma. Hasta donde se conoce, este es el primer estudio en el que se analiza dicho potencial.
- Se construye un nuevo conjunto de datos llamado Sohas_weapon (*small objects handled similarly to a weapon*) para el caso de estudio de seis objetos pequeños que suelen ser manipulados de forma similar a un arma: pistola, cuchillo, smartphone, billete, monedero y tarjeta. Se utilizaron diferentes tecnologías de cámaras de vigilancia para tomar las imágenes. El 10 % de las imágenes se descargaron de Internet. Todas estas imágenes fueron anotadas manualmente para la tarea de detección. Este conjunto de datos está disponible para otros estudios[2].

El estudio experimental realizado en la base de datos Sohas_weapon aplicando la metodología ODeBiC supera al modelo de detección de base hasta en un 19.57 % en precisión y reduce el número de falsos positivos hasta en un 56.50 %.

1.2. Trabajos relacionados con la detección de objetos en imágenes

Los trabajos relacionados con la detección de objetos en imágenes pueden dividirse en tres categorías: 1) trabajos previos que utilizan estrategias de binarización OVA y OVO en la clasificación, detección o segmentación, 2) el estado del arte de los modelos de detección de objetos en imágenes y 3) trabajos previos que abordan la detección de armas en vídeos.

La mayoría de los trabajos que analizan OVA y OVO en tareas visuales, reconocimiento de objetos, clasificación de imágenes y segmentación de imágenes, sólo utilizan modelos clásicos como *Support Vector Machine* (SVM), *Linear Discriminant Analysis* (LDA) y *k-Nearest Neighbours* (kNN):

- En la clasificación de imágenes, los autores en [RG14] analizaron el enfoque OVA y OVO para reducir el espacio de características en tres benchmarks de referencia conocidos, MNIST, *Amsterdam Library of Object Images* (ALOI) y *Australian Sign Language* (Auslan).
- Para la estimación de la pose utilizando la segmentación de imágenes, los autores en [YGK17] compararon un clasificador individual basado en CNN con OVA y OVO basado en SVM y mostraron que las CNN logran un rendimiento ligeramente mejor que OVA y OVO basado en SVM.
- De forma similar, en la tarea de clasificación de escenas en imágenes de teledetección, los autores en [CFHL15] también compararon OVA y OVO basados en SVM y 1-NN (*Nearest Neighbour*) y concluyeron que OVA proporcionaba peores resultados debido al desequilibrio entre clases. Los mejores resultados se obtuvieron con OVO basado en SVM.
- En el reconocimiento de caras, los autores en [ÖY17] utilizaron un modelo basado en CNN para la extracción de características y un SVM, OVA y OVO para la clasificación. Los mejores resultados se obtuvieron con una CNN combinada con SVM.
- Los autores en [LG05] compararon la técnica *Half-Against-Half* (HAH) con OVA y OVO en la clasificación de imágenes y encontraron que HAH proporciona resultados similares o peores en los benchmarks evaluados.

[2]Conjunto de datos Sohas_weapon: `http://sci2s.ugr.es/weapons-detection`

Por otro lado, hay que destacar que los modelos de detección del estado del arte son CNN *end-to-end* que combinan una meta-arquitectura de detección con un modelo de clasificación. Las meta-arquitecturas más influyentes son Faster-RCNN [RHGS15a], R-FCN [DRF16] y SDD [LAE+16]. Según [HRS+17], Faster-RCNN basado en Inception ResNet V2 obtiene la mayor precisión en objetos grandes, mientras que Faster-RCNN ResNet 101 proporciona la mayor precisión en objetos pequeños. SSD es el enfoque de detección más rápido, pero ofrece una precisión menor. El modelo que ofrece la mejor relación entre precisión y tiempo de ejecución es Faster-RCNN ResNet 101.

En videovigilancia, el primer modelo de detección de pistolas fue propuesto en [OTH18], proporcionando buenos resultados, pero produciendo un importante número de falsos positivos en el fondo debido a que el modelo confunde la pistola con objetos que se manejan de forma similar. Los autores en [OTC+19] proponen una técnica de fusión con el apoyo de dos cámaras simétricas para calcular el mapa de disparidad y luego restar el fondo y en consecuencia disminuir el número de falsos positivos en el fondo. En la misma dirección, los autores de [CTP+19] reducen el número de falsos positivos producidos por las condiciones extremas de luz utilizando un método de preprocesamiento guiado por la luminosidad.

Nuestro trabajo se diferencia de todos los citados anteriormente en que pretende desarrollar una metodología que reduzca el número de falsos positivos y mejore el rendimiento global en la detección de objetos pequeños manejados de forma similar. Como caso de estudio, se aborda el problema de la identificación de pequeños objetos manipulados de forma similar a un arma en vídeos de vigilancia. Este trabajo es el primero en aplicar OVA y OVO a modelos de aprendizaje profundo para la detección de objetos en imágenes y vídeos.

1.3. *One-Versus-All* (OVA)

La estrategia OVA [CB91, AMMR95] reformula el problema de clasificación multiclase en un conjunto de clasificadores binarios donde cada clasificador aprende a distinguir cada clase individual frente al resto de clases en conjunto. Este enfoque produce tantos clasificadores como el número de clases del problema original. La predicción final se calcula combinando las predicciones de los clasificadores individuales mediante un método de agregación denominado estrategia de *Maximum confidence strategy* (MAX). La clase con el mayor número de votos se considera la clase predicha. Formalmente, la regla de decisión MAX puede expresarse como,

$$Clase = arg \max_{i=1,\ldots,m} r_i, \tag{II.1}$$

donde $r_i \epsilon [0, 1]$ es la confianza para la clase i y m es el número de clases.

1.4. *One-Versus-One* (OVO)

La estrategia OVO [KPD90, PCST00, Abe03] traduce el problema original multiclase en tantos problemas binarios como combinaciones posibles entre pares de clases, para que cada clasificador aprenda a discriminar entre cada par. Es decir, un problema de m clases se convertirá en $m(m-1)/2$ clasificadores. En el caso concreto considerado en este trabajo con $m = 6$ se traducirán en 15 clasificadores. La predicción producida por todos los clasificadores se combinará en una matriz de confusión y se analizará mediante un método de agregación.

El sistema OVO puede utilizar diversas estrategias de agregación como, la regla *Max-Wins* (VOTE), *Weighted voting strategy* (WV), *Learning valued preference for classification* (LVPC), *Preference relations solved by Non-Dominance Criterion* (ND), *Classification by pairwise coupling* (PC), *Wu, Lin and Weng probability estimates by pairwise coupling approach* (PE) y *Distance-based relative competence weighting combination for OVO* (DRCW-OVO).

Regla *Max-Wins* (VOTE)

La regla VOTE, también llamada regla *Max-Wins* [Fri96], se considera la regla de decisión básica en OVO. Analiza cada elemento r_{ij} de la matriz de confusión, si la predicción r_{ij} es igual o mayor que 0.5, la clase de salida será i, por el contrario, la clase de salida será j. El resultado se suma y se selecciona la clase con más votos. Si hay dos o más clases con el mismo número de votos, se proponen dos alternativas:

- VOTE aleatorio: seleccionar uno al azar.
- VOTE por peso: sumar las predicciones y seleccionar como clase final la que obtiene el mayor valor.

Formalmente, la regla de decisión se puede escribir como:

$$Clase = arg \max_{i=1,\ldots,m} \sum_{i \leq j \neq i \leq m} s_{ij}, \tag{II.2}$$

donde s_{ij} es 1 si $r_{ij} > r_{ji}$ y 0 en caso contrario.

***Weighted voting strategy* (WV)**

El objetivo de esta técnica [GFB+11] es obtener la clase con mayor probabilidad. Para ello, cada clase suma sus predicciones y la clase con el valor mayor será el resultado final. La regla de decisión es:

$$Clase = arg \max_{i=1,\ldots,m} \sum_{i \leq j \neq i \leq m} r_{ij} \tag{II.3}$$

***Learning valued preference for classification* (LVPC)**

Learning valued preference for classification (LVPC) calcula algunos valores nuevos a partir de las probabilidades iniciales obtenidas por los clasificadores binarios. LVPC es una votación ponderada, que penaliza a los clasificadores que no tienen un umbral de confianza en su decisión. Más detalles sobre esta regla se encuentran en [HB08, HH09]. Esta regla de decisión puede expresarse como:

$$\begin{aligned} P_{ij} &= r_{ij} - min\{r_{ij}, r_{ji}\} \\ P_{ji} &= r_{ji} - min\{r_{ij}, r_{ji}\} \\ C_{ij} &= min\{r_{ij}, r_{ji}\} \\ I_{ij} &= 1 - max\{r_{ij}, r_{ji}\} \\ Clase &= arg \max_{i=1,\ldots,m} \sum_{i \leq j \neq i \leq m} P_{ij} + \frac{1}{2} C_{ij} + \frac{N_i}{N_i + N_j} I_{ij}, \end{aligned} \tag{II.4}$$

donde N_i es el número de ejemplos de la clase i en los datos de entrenamiento.

***Preference relations solved by Non-Dominance Criterion* (ND)**

La técnica ND, también llamada, *Preference relations solved by Non-Dominance Criterion*, se introdujo inicialmente en la toma de decisiones con relaciones de preferencia difusas [Orl78, FCB+09]. El mismo criterio puede aplicarse a un sistema de clasificación OVO.

En primer lugar, se debe normalizar:

$$\overline{r}_{ij} = \frac{r_{ij}}{r_{ij} + r_{ji}} \tag{II.5}$$

A continuación, se calcula la preferencia:

$$r'_{ij} = \begin{cases} \overline{r}_{ij} - \overline{r}_{ji}, & donde\, \overline{r}_{ij} > \overline{r}_{ji} \\ 0, & deotraforma. \end{cases} \tag{II.6}$$

Y, se calcula el grado de no dominancia de cada clase:

$$ND_i = 1 - \sup_{j \epsilon C}[r'_{ji}] \tag{II.7}$$

Finalmente, la salida será:

$$ClasePredicha = arg \max_{i=1,\ldots,m} ND_i \tag{II.8}$$

***Classification by pairwise coupling* (PC)**

La técnica PC o *Classification by Pairwise coupling* [HT98] intenta mejorar la estrategia de votación cuando las salidas de los clasificadores son probabilidades. Este método calcula la probabilidad conjunta de todas las clases a partir de las probabilidades de clase por pares de los clasificadores binarios.

El algoritmo propuesto es:

1. Inicialización:

$$\begin{aligned} \widehat{p}_i &= \frac{2}{m}\frac{\sum\limits_{1\leq j\neq i\leq m} r_{ij}}{(m-1)} para\ todo\ i = 1, ..., m \\ \widehat{\mu}_{ij} &= \frac{\widehat{p}_i}{\widehat{p}_i + \widehat{p}_j} para\ todo\ i, j = 1, ..., m \end{aligned} \tag{II.9}$$

2. Repetir hasta converger:

   a) Calcular $\widehat{p}$

$$\widehat{p}_i = \widehat{p}_i \frac{\sum\limits_{1\leq j\neq i\leq m} n_{ij} r_{ij}}{\sum\limits_{1\leq j\neq i\leq m} n_{ij}\widehat{\mu}_{ij}} para\ todo\ i = 1, ..., m \tag{II.10}$$

   donde n_{ij} es el número de datos de entrenamiento en las clases ith y jth.

   b) Normalizar $\widehat{p}$

$$\widehat{p}_i = \frac{\widehat{p}_i}{\sum\limits_{i=1}\widehat{p}_i} para\ todo\ i = 1, ..., m \tag{II.11}$$

   c) Recalcular $\widehat{\mu}_{ij}$

$$\widehat{\mu}_{ij} = \frac{\widehat{p}_i}{\widehat{p}_i + \widehat{p}_j} para\ todo\ i, j = 1, ..., m \tag{II.12}$$

Por último, la clase de salida:

$$Clase = arg \max_{i=1,...,m} \widehat{p}_i \tag{II.13}$$

***Wu, Lin and Weng probability estimates by pairwise coupling approach* (PE)**

La técnica PE, también llamada probabilidad de Wu, Lin y Weng, es similar a la técnica PC. Utiliza el enfoque de acoplamiento por pares para calcular las predicciones [WLW04]. Las probabilidades (p) de cada clase se estiman a partir de las probabilidades por pares. PE optimiza el siguiente problema:

$$\min_{p} \sum_{i=1}^{m} \sum_{1\leq j\neq i\leq m} (r_{ji}p_i - r_{ij}p_j)^2\ sujeto\ a \sum_{i=1}^{k} p_i = 1, p_i \geq 0, \forall i \tag{II.14}$$

***Distance-based relative competence weighting combination for One-Versus-One* (DRCW-OVO)**

Distance-based relative competence weighting combination, también llamada estrategia OVO en problemas multiclase (DRCW-OVO) [GFBH15], es una de las variaciones [CSC18] de la técnica OVO que pretende mejorar el problema de las clases desequilibradas utilizando la distancia con los k elementos cercanos a la nueva instancia.

Una vez obtenida la matriz de puntuación, DRCW-OVO implica lo siguiente:

1. Calcular la distancia media de los k vecinos más cercanos de cada clase en un vector **d**.
2. Calcular la nueva matriz de puntuación R^w de la siguiente manera:

$$r_{ij}^w = r_{ij} \cdot w_{ij}, \tag{II.15}$$

donde w_{ij} se calcula como:

$$w_{ij} = \frac{d_j^2}{d_i^2 + d_j^2}, \tag{II.16}$$

siendo d_i la distancia de la instancia al vecino más cercano de la clase i.

3. Utilizar la estrategia de votación ponderada (WV) sobre la nueva matriz de puntuación R^w para obtener la clase final.

Al trabajar con imágenes en este problema, se necesita calcular la distancia y para ello, una forma de hacerlo es calcular la Chi Cuadrática [PW10] con el histograma de las imágenes:

$$X^2(P,Q) = \frac{1}{2}\sum_i \frac{(P_i - Q_i)^2}{(P_i + Q_i)}, \tag{II.17}$$

donde P_i es el histograma de la nueva instancia y Q_i es la media del histograma de los k vecinos más cercanos.

2. Base de datos Sohas_weapon y metodología ODeBiC basada en *deep learning*

En esta sección se propone la metodología ODeBiC basada en *deep learning* para clasificadores binarios con el objetivo de detectar objetos pequeños que pueden ser confundidos porque se manejan de forma similar. Como caso de estudio, se selecciona un problema del ámbito de la videovigilancia, la detección de armas. Además, se crea el conjunto de datos llamado Sohas_weapon.

En esta sección, primero se describe el proceso realizado para construir un conjunto de datos de objetos pequeños que se pueden manejar de forma similar a un arma (subsección 2.1). A continuación, se presenta la metodología ODeBiC (subsección 2.2).

2.1. Construcción de la base de datos Sohas_weapon para detección en videovigilancia

La calidad del aprendizaje de un modelo CNN depende en gran medida de la calidad de la base de datos de entrenamiento. La base de datos debe permitir que el modelo de clasificación distinga correctamente entre objetos que se manejan de forma similar.

Se han construido cuatro bases de datos para el entrenamiento de los modelos de clasificación, Database-1, 2, 3 y 4, utilizando diferentes tipos de imágenes. Estas bases de datos se basan en objetos que se manipulan de forma similar, como pistola, cuchillo, smartphone, billete, monedero y tarjeta:

1. En el primer paso, se utilizaron las imágenes de pistolas de la base de datos[3] construida en [OTH18] y las imágenes de cuchillos de la base de datos[4] construida en [CTP+19], donde la mayoría de las imágenes fueron descargadas de Internet. A estos datos se añadieron las imágenes de objetos comunes que se pueden manejar de forma similar a una pistola y un cuchillo, como smartphone, billete, monedero y tarjeta. Esta base de datos se llama Database-1.
2. En un segundo paso, se añadieron a cada clase imágenes tomadas en diversas condiciones por una cámara réflex, Nikon D5200. La base de datos obtenida se llama Database-2.
3. En un tercer paso, se añadieron a cada clase imágenes tomadas por dos cámaras de vigilancia con diferentes calidades y resoluciones, Hikvision DS-2CD2420F-IW y Samsung SNH-V6410PN, y en diversas condiciones. La base de datos obtenida se denomina Database-3.
4. En el último paso, se eliminaron las imágenes borrosas debido al movimiento y las imágenes en las que el ojo humano no puede reconocer la clase del objeto. La base de datos final se llama Sohas_weapon.

Para evaluar la calidad de las bases de datos, guiadas por la calidad del aprendizaje de los enfoques de clasificación, se construyó una base de datos llamada Database-Sohas_weapon-Test. Las características de todas las bases de datos construidas se proporcionan en la tabla II.1. Además, se utilizó una base de datos sin la clase pistola ni cuchillo, Database-Sohas_weapon-Without_Pistol&Knife y Database-Sohas_weapon-Test_Without_Pistol&Knife, para analizar el comportamiento de los enfoques de clasificación propuestos sobre los objetos que tienen una mayor similitud en la forma y el modo de manipulación, smartphone, billete, monedero y tarjeta.

Para el entrenamiento de los modelos de detección, se utilizó la base de datos Database-Sohas_weapon-Detection cuyas características se resumen en la tabla II.1. Esta base de datos contiene las imágenes completas (objetos y fondo) de las que se recortaron las imágenes utilizadas para construir la base de datos Sohas_weapon.

Para analizar la metodología ODeBiC, se crearon cuatro vídeos de vigilancia para test cuyas características se resumen en la tabla II.2. Estos cuatro vídeos de vigilancia se grabaron en diferentes escenarios: en una pequeña oficina y en la entrada de un edificio, con la cámara Samsung SNH-V6410PN.

2.2. Metodología ODeBiC basada en *deep learning*

Uno de los principales problemas en la detección de objetos en vídeos de vigilancia es que los objetos que son manejados de forma similar pueden confundirse. Esto se demostró en el modelo de detección de

[3]Base de datos de 3000 pistolas: `http://sci2s.ugr.es/weapons-detection`

[4]Base de datos de cuchillos: `https://github.com/ari-dasci/OD-WeaponDetection/tree/master/Knife_detection`

Tabla II.1: Base de datos construida para analizar el rendimiento de los objetos que se manipulan de forma similar a un arma.

Database-	**# img**	**Pistola**	**Cuchillo**	**Smartphone**	**Billete**	**Monedero**	**Tarjeta**
1	6589	3394	1879	866	134	137	179
2	7333	3523	1879	1022	287	315	307
3	8537	3681	1879	1069	654	710	544
Sohas_weapon	5680	1580	1879	755	545	581	340
Sohas_weapon-Without_Pistol&Knife	2221	0	0	755	545	581	340
Sohas_weapon-Detection	5080	1425	1825	575	425	530	300
Sohas_weapon-Test	1170	294	470	115	123	104	64
Sohas_weapon-Test_Without_Pistol&Knife	406	0	0	115	123	104	64

Tabla II.2: Cuatro vídeos de vigilancia de test creados para analizar el rendimiento de la metodología ODeBiC.

Vídeo	**# Frames**	**Pistola**	**Cuchillo**	**Smartphone**	**Billete**	**Monedero**	**Tarjeta**	**Escenario**
1	1962	235	289	217	302	342	391	Pequeña oficina
2	2083	269	256	477	282	294	417	Entrada, alejado
3	2070	329	274	284	294	330	356	Entrada, cercano
4	2188	315	246	458	323	331	504	Entrada, pasillo

pistolas desarrollado en el trabajo inicial [OTH18].

En este trabajo, se propone el uso de la metodología ODeBiC basada en *deep learning* para mejorar la fiabilidad, la robustez y la precisión para identificar objetos pequeños manipulados de forma similar a un arma. La metodología ODeBiC tiene dos niveles, en el primero se obtienen las regiones candidatas que contienen los objetos objetivo, y en el segundo se clasifica cada región con la técnica de binarización seguida de un método de agregación para finalmente producir el frame de salida con los resultados de detección. En concreto, la metodología ODeBiC funciona de la siguiente forma:

- En el primer nivel se analiza el fotograma de entrada mediante un modelo de detección CNN relajado que produce todas las propuestas de regiones con una probabilidad de tener uno o más objetos objetivo superior al 10 %. Este proceso podría considerarse como una técnica de selección de candidatos con un importante conocimiento de las categorías de objetos objetivo. Consideraremos Faster-RCNN basado en el extractor de características ResNet 101, ya que ofrece un buen equilibrio entre precisión y tiempo de ejecución. Estos candidatos serán analizados por el segundo nivel.
- Cada caja de salida será analizada por una técnica de binarización, y seguidamente se aplicará un método de agregación para calcular la predicción final. Se considerarán dos técnicas de binarización, OVA y OVO, en combinación con diferentes métodos de agregación. Una ilustración de OVA y OVO en el contexto del problema de pistola, cuchillo y objetos manejados similarmente se representa en la figura 24 y en la figura 25 respectivamente.

La metodología de dos niveles propuesta se representa en la figura 26.

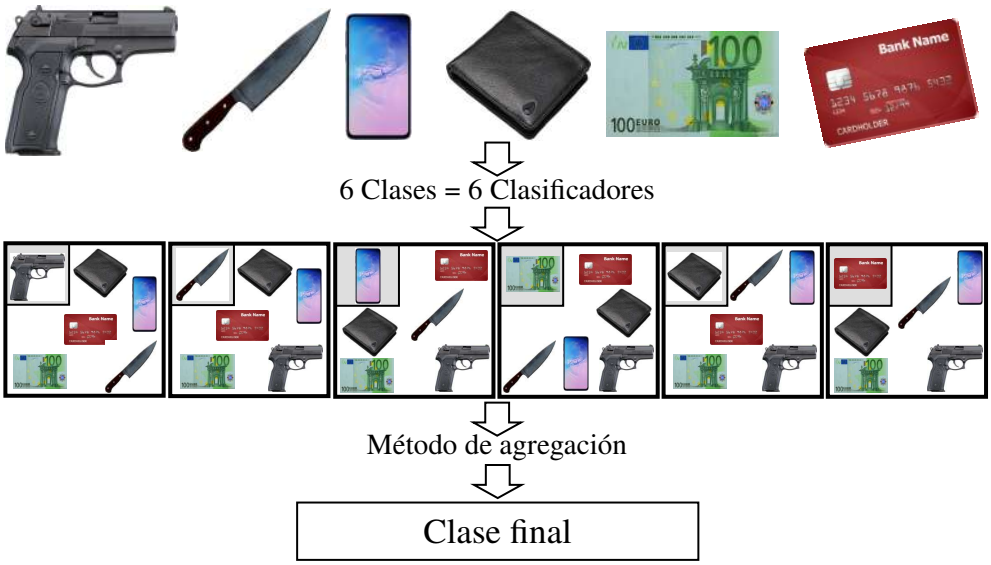


Figura 24: OVA para el reconocimiento de objetos pequeños que pueden ser manejados de forma similar.

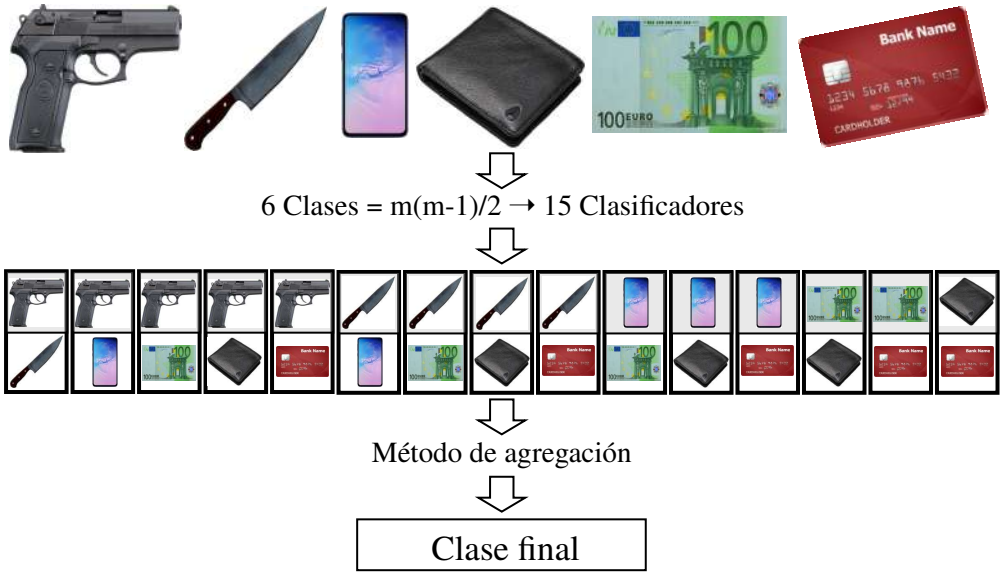


Figura 25: OVO para el reconocimiento de objetos pequeños que pueden ser manejados de forma similar.

3. Estudio experimental

El propósito de esta sección es analizar el rendimiento de diferentes enfoques de clasificación, el multiclasificador de base, OVA y OVO con varias reglas de agregación en la subsección 3.1, el estudio de los objetos similares en la subsección 3.2 y la evaluación de la metodología ODeBiC utilizando cuatro vídeos de vigilancia en la subsección 3.3.

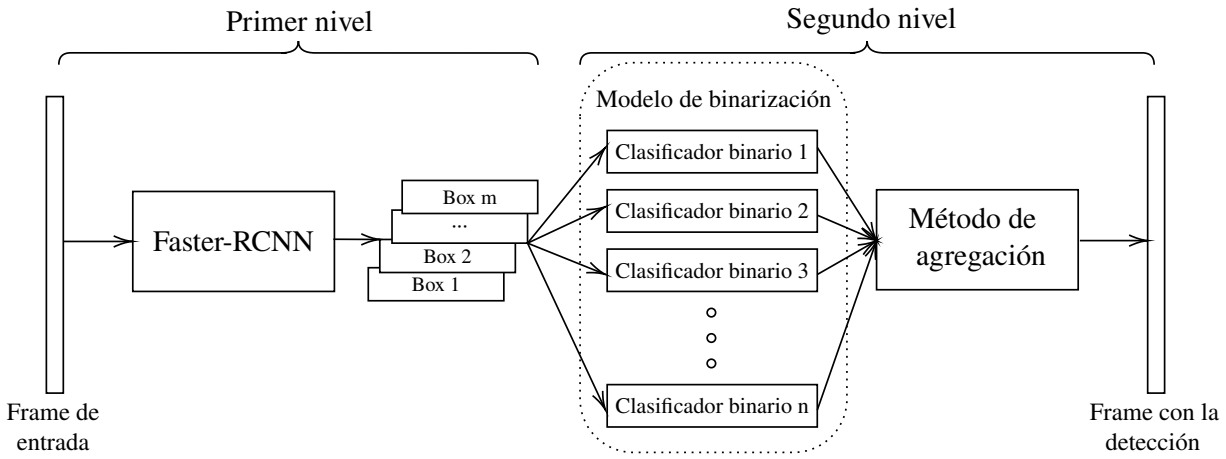


Figura 26: Metodología ODeBiC de dos niveles, primero detección y a continuación binarización.

3.1. Evaluación de diferentes aproximaciones de clasificación

En esta subsección se analiza el rendimiento de diferentes enfoques de clasificación, el multiclasificador de base, OVA y OVO con diferentes reglas de agregación, VOTE aleatorio, VOTE por peso, WV, LVPC, ND, PC, PE, DRCW con k=1, 2, 3 y 4, entrenados en Database-1, 2, 3 y Sohas_weapon-Test. Todos los modelos CNN analizados se basan en la arquitectura ResNet 101 [HZRS16] inicializada con los pesos pre-entrenados en ImageNet [DDS$^+$09]. Se utiliza TensorFlow [ABC$^+$16] y NVIDIA Titan Xp para todos los experimentos. El proceso de entrenamiento dura aproximadamente dos horas. Los resultados se presentan en la figura 27 y se resumen en la tabla II.3.

Como se puede observar en la figura 27, en general, el rendimiento de todos los enfoques aumenta desde Database-1 hasta Database-Sohas_weapon. En particular, cuando se entrenan en la Database-1, OVA y OVO proporcionan un rendimiento similar al del multiclasificador base. En Database-2, OVA obtiene el mejor rendimiento de todos los métodos. En Database-3 y Database-Sohas_weapon, todos los métodos de agregación de OVO proporcionan un mejor rendimiento que el multiclasificador base.

DRCW-OVO con k=1 obtiene los mejores resultados en Database-3. En Database-Sohas_weapon, OVO ND proporciona los mejores resultados con una precisión del 93.87 %, un recall del 93.09 % y un F1 del 93.43 %. La mejora con respecto al multiclasificador de base en Database-Sohas_weapon es del 2.57 % en precisión, 2.06 % en recall y 2.34 % en F1. Sin embargo, en términos de tiempo de ejecución, DRCW-OVO tarda 4.04 segundos por fotograma, ya que para cada imagen de entrada calcula la distancia entre todas las imágenes de la base de datos. Esto hace que DRCW-OVO no sea apropiado para el procesamiento en tiempo real. Por lo tanto, para evaluar la propuesta, se seleccionan sólo los modelos que proporcionan una buena relación precisión/tiempo de ejecución, OVO con diferentes reglas de agregación, VOTE aleatorio, VOTE por peso, WV, LVPC, ND, PC y PE.

Como conclusión de esta evaluación, el uso de técnicas de binarización produce mejores resultados que el multiclasificador de base. Además, este tipo de técnicas pueden ser utilizadas en tiempo real a excepción de DRCW-OVO.

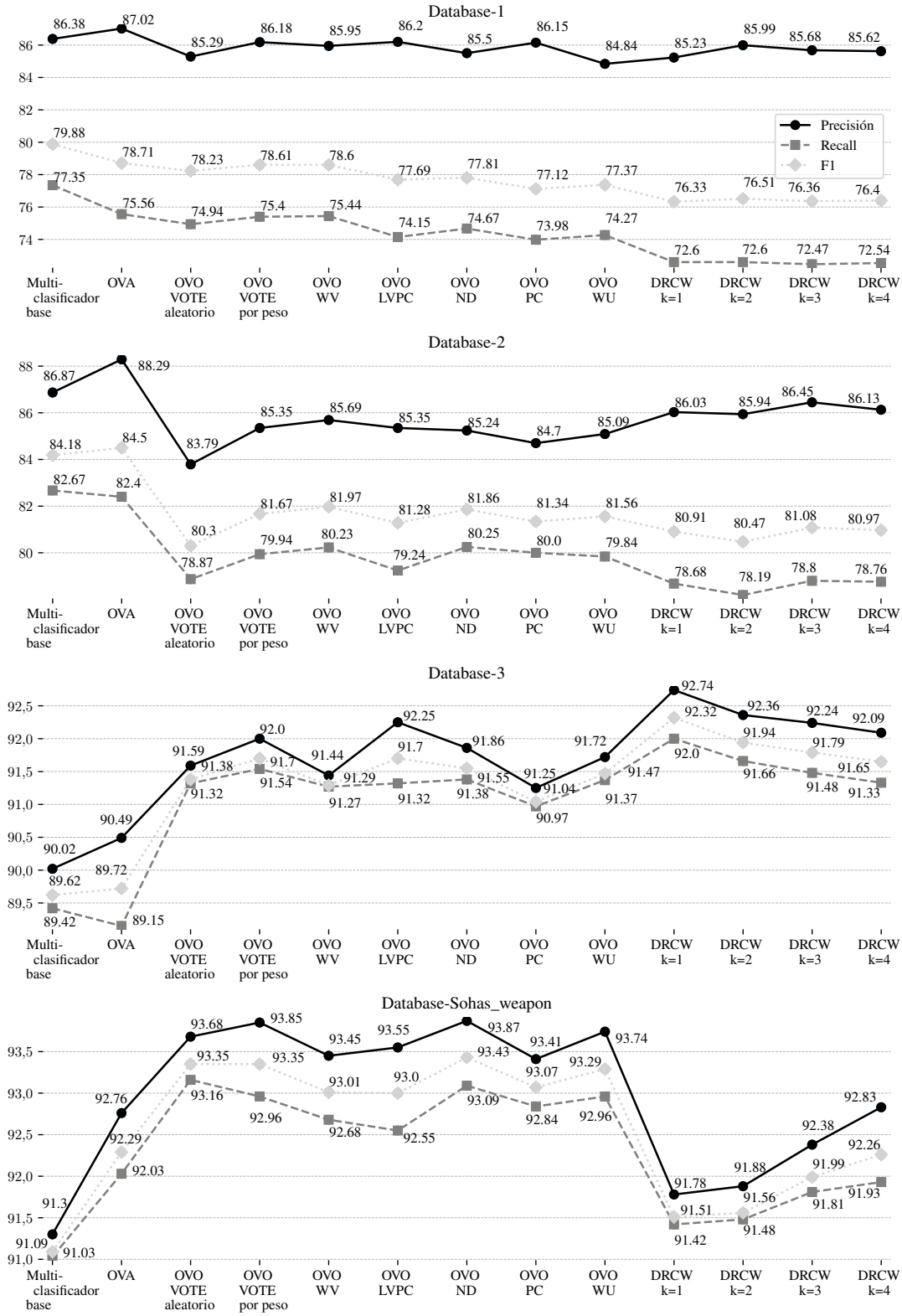


Figura 27: Resultados, en %, de cada enfoque de clasificación entrenado en Database-1, 2, 3 y Sohas_weapon y testeado en Database-Sohas_weapon-Test.

Tabla II.3: Resultados, en %, de todos los enfoques de clasificación entrenados en Database-1, 2, 3 y Sohas_weapon y testados en Database-Sohas_weapon-Test.

	Database-1			**Database-2**			**Database-3**			**Database-Sohas_weapon**			**Tiempo (s)**
	Precisión	Recall	F1	Precisión	Recall	F1	Precisión	Recall	F1	Precisión	Recall	F1	
Multiclasificador base	86.38	77.35	79.88	86.87	82.67	84.18	90.02	89.42	89.62	91.30	91.03	91.09	0,02821
OVA	87.02	75.56	78.71	88.29	82.40	84.50	90.49	89.15	89.72	92.76	92.03	92.29	0,03081
OVO VOTE aleatorio	85.29	74.94	78.23	83.79	78.87	80.30	91.59	91.32	91.38	93.68	**93.16**	93.35	0,02824
OVO VOTE por peso	86.18	75.40	78.61	85.35	79.94	81.67	92.00	91.54	91.70	93.85	92.96	93.35	0,02823
OVO WV	85.95	75.44	78.60	85.69	80.23	81.97	91.44	91.27	91.29	93.45	92.68	93.01	0,02822
OVO LVPC	86.20	74.15	77.69	85.35	79.24	81.28	92.25	91.32	91.70	93.55	92.55	93.00	0,02828
OVO ND	85.50	74.67	77.81	85.24	80.25	81.86	91.86	91.38	91.55	**93.87**	93.09	**93.43**	0,02827
OVO PC	86.15	73.98	77.12	84.70	80.00	81.34	91.25	90.97	91.04	93.41	92.84	93.07	0,04493
OVO PE	84.84	74.27	77.37	85.09	79.84	81.56	91.72	91.37	91.47	93.74	92.96	93.29	0,02830
DRCW k=1	85.23	72.60	76.33	86.03	78.68	80.91	92.74	92.00	92.32	91.78	91.42	91.51	4,02127
DRCW k=2	85.99	72.60	76.51	85.94	78.19	80.47	92.36	91.66	91.94	91.88	91.48	91.56	4,02127
DRCW k=3	85.68	72.47	76.36	86.45	78.80	81.08	92.24	91.48	91.79	92.38	91.81	91.99	4,02127
DRCW k=4	85.62	72.54	76.40	86.13	78.76	80.97	92.09	91.33	91.65	92.83	91.93	92.26	4,02127

3.2. Objetos similares: Análisis

El objetivo de esta subsección es estudiar el comportamiento de las técnicas de binarización en problemas más difíciles cuando la similitud entre todos los objetos de la base de datos es mayor, como en el caso del smartphone, billete, monedero y tarjeta. Para ello, se elimina la clase pistola y cuchillo de Database-Sohas_weapon y Database-Sohas_weapon-Test obteniendo Database-Sohas_weapon-Without_Pistol&Knife y Database-Sohas_weapon-Test_Without_Pistol&Knife.

El rendimiento de todos los enfoques analizados, el multiclasificador de base, OVA y OVO con diversas reglas de agregación, VOTE aleatorio, VOTE por peso, WV, LVPC, ND, PC, PE, DRCW con k=1, 2, 3 y 4, cuando se entrenan en Database-Sohas_weapon-Without_Pistol&Knife y se testean en Database-Sohas_weapon-Test_Without_Pistol&Knife y el resultado se proporciona en la tabla II.4.

Como se observa en la tabla II.4, DRCW-OVO k=1 consigue los mejores valores medios de Precisión, Recall y F1, con los valores respectivos de 93.76 %, 93.46 % y 93.48 %. Sin embargo, estos resultados fueron similares con respecto al estudio que incluye pistola y cuchillo. Para un análisis más detallado, la tabla II.5 muestra las matrices de confusión del estudio con y sin pistola y cuchillo con su mejor método de agregación, el que obtiene el mayor rendimiento, OVO-ND.

Como se puede observar en la tabla II.5, la precisión media, recall y F1 tienen un valor similar con respecto al caso de la pistola y el cuchillo debido a que la clase de pistola y cuchillo alcanzan el mayor rendimiento sobre el resto de los objetos. Esto puede explicarse por el desequilibrio de la base de datos y también por la alta calidad y cantidad de las imágenes de pistola y cuchillo.

DRCW-OVO con k=1 entrenado en Database-Sohas_weapon-Without_Pistol&Knife comete menos errores en la mayoría de los objetos similares, billete, monedero, smartphone y tarjeta. Como resumen, el enfoque de binarización en general y DRCW-OVO con k=1 en particular ayudan a diferenciar correctamente los objetos similares.

Tabla II.4: Resultado, en %, de todos los enfoques de clasificación entrenados en Database-Sohas_weapon-Without_Pistol&Knife y testeados en Database-Sohas_weapon-Test_Without_Pistol&Knife.

Database-Sohas_weapon-Without_Pistol&Knife			
	Precisión	**Recall**	**F1**
Multiclasificador base	91.27	90.46	90.63
OVA	91.70	91.28	91.32
OVO VOTE aleatorio	92.63	92.69	92.62
OVO VOTE por peso	93.51	93.41	93.39
OVO WV	93.29	93.20	93.18
OVO LVPC	93.07	92.81	92.87
OVO ND	93.28	93.02	93.08
OVO PC	93.29	93.20	93.18
OVO PE	93.07	92.81	92.87
DRCW-OVO k=1	**93.76**	**93.46**	**93.48**
DRCW-OVO k=2	93.22	92.95	93.03
DRCW-OVO k=3	93.02	92.75	92.82
DRCW-OVO k=4	93.02	92.75	92.82

Tabla II.5: Matriz de confusión de la mejor técnica en Database-Sohas_weapon y Database-Sohas_weapon-Without_Pistol&Knife, OVO-ND y DRCW-OVO k=1 respectivamente.

Database-Sohas_weapon									
	Billete	**Cuchillo**	**Monedero**	**Pistola**	**Smartphone**	**Tarjeta**	**Precisión**	**Recall**	**F1**
Billete	118	1	0	0	0	2	97.52	95.93	96.72
Cuchillo	0	457	2	2	4	0	98.28	97.23	97.75
Monedero	3	0	94	3	10	0	85.45	90.38	87.85
Pistola	0	10	4	289	1	3	94.14	98.30	96.17
Smartphone	0	1	4	0	99	1	94.29	86.09	90.00
Tarjeta	2	1	0	0	1	58	93.55	90.63	92.06
							93.87	93.09	93.43

Database-Sohas_weapon-Without_Pistol&Knife									
	Billete	**Cuchillo**	**Monedero**	**Pistola**	**Smartphone**	**Tarjeta**	**Precisión**	**Recall**	**F1**
Billete	120	-	1	-	0	2	97.56	97.56	97.56
Monedero	2	-	101	-	13	0	87.07	97.12	91.82
Smartphone	0	-	2	-	100	3	95.24	86.96	90.91
Tarjeta	1	-	0	-	2	59	95.16	92.19	93.65
							93.76	93.46	93.48

Este estudio de objetos similares demuestra cómo las técnicas de binarización aumentan el rendimiento en situaciones difíciles en las que OVO, con métodos de agregación múltiple, obtiene un mejor rendimiento.

3.3. Evaluación de la metodología ODeBic en vídeos de vigilancia

En esta sección se analiza la metodología ODeBiC utilizando cuatro vídeos de vigilancia descritos en la tabla II.2.

Para el entrenamiento de los modelos de detección, se utiliza la Database-Sohas_weapon-Detection cuyas características se resumen en la tabla II.1. Se considerará en este análisis sólo los modelos de clasificación que son apropiados para la ejecución en tiempo real, OVO con diferentes reglas de agregación, VOTE aleatorio, VOTE por peso, WV, LVPC, ND, PC y PE.

Para el primer paso de la metodología ODeBiC, se utiliza Faster-RCNN basado en ResNet 101 entrenado en Database-Sohas_weapon-Detection. En esta fase, el modelo de detección analiza los vídeos fotograma a fotograma y obtiene las zonas con una confianza de detección superior a un umbral mínimo. Estas regiones propuestas serán analizadas por un método de binarización en la segunda etapa mediante técnicas de OVO. A efectos de comparación, se utiliza Faster-RCNN basado en ResNet 101 como detector de base.

La tabla II.6 muestra el rendimiento de la metodología ODeBiC cuando se utilizan diferentes umbrales, 10 %, 50 %, 70 % y 90 %, en el primer nivel de la metodología ODeBiC. El umbral se refiere a la confianza del modelo en la detección de los objetos considerados. Para el segundo nivel, se ha considerado la técnica de binarización OVO con diferentes métodos de agregación, VOTE aleatorio, VOTE por peso, WV, LVPC, ND, PC y PE. El número real de pistolas, cuchillos y objetos similares en cada vídeo se indica como número de GT (*Ground Truth*).

En general, como se puede observar en la tabla II.6, la metodología propuesta basada en la técnica OVO con todos los métodos de agregación supera al modelo de detección de base en los vídeos analizados. Los mejores resultados se obtuvieron con OVO con el método de agregación PC o WV en los vídeos, y en todos los valores de umbral.

Los resultados pueden resumirse como sigue:

- La metodología ODeBiC basada en el método de agregación OVO supera al modelo de base en precisión entre el 10.68 % y el 19.57 % para el umbral del 10 %, entre el 4.81 % y el 12.24 % para el umbral del 50 %, entre el 2.44 % y el 9.77 % para el umbral del 70 % y entre el -2.19 % y el 5.88 % para el umbral del 90 %.

- En cuanto a los falsos positivos, se reduce el número de FP entre el 34.64 % y el 56.50 % para un umbral del 10 %, entre el 22.14 % y el 47.39 % para un umbral del 50 %, entre el 12.76 % y el 43.01 % para un umbral del 70 % y entre el -16.54 % y el 33.89 % para un umbral del 90 %.

- En términos de tiempo de ejecución, el modelo de detección de base tarda 0.12341 segundos (equivalente a 8 fps) y la metodología ODeBiC con OVO PC tarda alrededor de 0.16834 (equivalente a 6 fps), lo que es apropiado para un sistema aproximado de tiempo real.

En resumen, la metodología ODeBiC se ejecuta casi en tiempo real y consigue una mejora de hasta el 56.50 % utilizando un método de agregación de OVO.

Tabla II.6: Resultados, en %, de la metodología ODeBiC en cuatro vídeos de vigilancia.

		Umbral 10 %			Umbral 50 %			Umbral 70 %			Umbral 90 %		
		TP	**FP**	**Precisión**	**TP**	**FP**	**Precisión**	**TP**	**FP**	**Precisión**	**TP**	**FP**	**Precisión**
Vídeo 1 1776 GT	Base	1189	630	65.37	994	346	74.18	926	272	77.30	843	177	82.65
	OVO VOTE aleatorio	1540	279	84.66	1156	184	86.27	1037	161	86.56	900	120	88.24
	OVO VOTE por peso	1535	284	84.39	1150	190	85.82	1035	163	86.39	898	122	88.04
	OVO WV	1545	274	**84.94**	1158	182	**86.42**	1043	155	**87.06**	903	117	**88.53**
	OVO LVPC	1529	290	84.06	1148	192	85.67	1037	161	86.56	900	120	88.24
	OVO ND	1535	284	84.39	1150	190	85.82	1036	162	86.48	898	122	88.04
	OVO PC	1525	294	83.84	1137	203	84.85	1022	176	85.31	882	138	86.47
	OVO PE	1533	286	84.28	1155	185	86.19	1037	161	86.56	899	121	88.14
Vídeo 2 1995 GT	Base	1248	617	66.92	1064	332	76.22	992	235	80.85	870	133	**86.74**
	OVO VOTE aleatorio	1368	497	73.35	1084	312	77.65	972	255	79.22	821	182	81.85
	OVO VOTE por peso	1385	480	74.26	1094	302	78.37	981	246	79.95	826	177	82.35
	OVO WV	1402	463	75.17	1101	295	78.87	985	242	80.28	828	175	82.55
	OVO LVPC	1367	498	73.30	1078	318	77.22	970	257	79.05	818	185	81.56
	OVO ND	1380	485	73.99	1091	305	78.15	978	249	79.71	823	180	82.05
	OVO PC	1480	385	**79.36**	1148	248	**82.23**	1022	205	**83.29**	848	155	84.55
	OVO PE	1378	487	73.89	1092	304	78.22	977	250	79.63	825	178	82.25
Vídeo 3 1867 GT	Base	1250	557	69.18	1073	298	78.26	1014	241	80.80	901	158	85.08
	OVO VOTE aleatorio	1403	404	77.64	1116	255	81.40	1041	214	82.95	911	148	86.02
	OVO VOTE por peso	1417	390	78.42	1127	244	82.20	1051	204	83.75	917	142	**86.59**
	OVO WV	1421	386	78.64	1126	245	82.13	1049	206	83.59	914	145	86.31
	OVO LVPC	1406	401	77.81	1118	253	81.55	1043	212	83.11	910	149	85.93
	OVO ND	1409	398	77.97	1120	251	81.69	1045	210	83.27	913	146	86.21
	OVO PC	1443	364	**79.86**	1139	232	**83.08**	1056	199	**84.14**	912	147	86.12
	OVO PE	1407	400	77.86	1118	253	81.55	1044	211	83.19	914	145	86.31
Vídeo 4 2177 GT	Base	1502	742	66.93	1301	381	77.35	1211	292	80.57	1063	208	83.63
	OVO VOTE aleatorio	1816	428	80.93	1404	278	83.47	1266	237	84.23	1093	178	86.00
	OVO VOTE por peso	1819	425	81.06	1409	273	83.77	1270	233	84.50	1096	175	86.23
	OVO WV	1821	423	81.15	1409	273	83.77	1269	234	84.43	1094	177	86.07
	OVO LVPC	1794	450	79.95	1392	290	82.76	1253	250	83.37	1083	188	85.21
	OVO ND	1819	425	81.06	1408	274	83.71	1269	234	84.43	1095	176	86.15
	OVO PC	1874	370	**83.51**	1440	242	**85.61**	1296	207	**86.23**	1113	158	**87.57**
	OVO PE	1807	437	80.53	1397	285	83.06	1260	243	83.83	1087	184	85.52

4. Conclusiones

Este trabajo presenta la metodología de dos niveles ODeBiC basada en *deep learning* para la detección de objetos pequeños que pueden ser manejados de forma similar a un arma. Se considera como caso de estudio la detección de objetos pequeños que pueden confundirse con una pistola o un cuchillo en vídeos de vigilancia. Se construye una base de datos de entrenamiento, llamada Sohas_weapon, que incluye seis objetos que pueden confundirse con un arma, ya que se manejan comúnmente de forma similar: pistola, cuchillo, smartphone, billete, monedero y tarjeta.

Los experimentos mostraron que la metodología ODeBiC, basada en un método de agregación de OVO, redujo el número de falsos positivos hasta en un 56.50 % y entre un -2.19 % y un 19.57 % en precisión, dependiendo del umbral, con respecto al modelo de detección de base.

La metodología ODeBiC puede utilizarse como ayuda para la videovigilancia, ya que produce resultados robustos, reduce considerablemente el número de falsos positivos y obtiene una mayor precisión que el modelo de detección de base.

Los resultados de la investigación realizada en esta etapa del trabajo fueron presentados en un articulo aceptado por la revista *Knowledge-Based Systems*.

Capítulo III

Metodología DetDSCI para la detección de infraestructuras críticas en imágenes de satélite

En este capítulo se propone el desarrollo de una metodología basada en una técnica de preprocesamiento de modelos *deep learning* para abordar la detección de infraestructuras críticas en imágenes de satélite. Este problema es de suma importancia ya que esta herramienta se puede utilizar para reconocer terrenos inexplorados.

La detección de infraestructuras críticas es un problema complejo ya que la principal limitación en este problema consiste en que las infraestructuras tienen tamaños muy dispares. Algunas como las subestaciones eléctricas ocupan una superficie del orden de varios m^2 mientras que infraestructuras como los aeropuertos pueden tener una superficie de hasta decenas de Km^2. Además, del problema de intra-variabilidad, la misma infraestructura puede tener aspectos muy diferentes como es el aeropuerto de Almería que presenta una forma muy diferente al aeropuerto de Barcelona.

Para atajar este problema se necesita realizar la construcción de una base de datos especializada en estas infraestructuras a través de las imágenes de satélite de fuentes gratuitas como Google Maps. Estas imágenes serán usadas por modelos de detección de objetos que proporcionan grandes desempeños en esta tarea.

Para abordar el problema de la detección de infraestructuras se ha diseñado la metodología DetDSCI con la que se hace uso de una técnica de preprocesamiento de modelos *deep learning* obteniendo mejores resultados que los modelos base. Esta metodología está compuesta de dos fases donde la primera de ellas es la encargada de seleccionar el grupo de nivel de zoom al que pertenece la imagen de entrada. Una vez que se ha seleccionado el intervalo al que pertenece se selecciona un detector experto en el rango de niveles de zoom para realizar la detección de infraestructuras en la imagen.

El uso de la metodología DetDSCI para el problema de la detección de infraestructuras críticas supone una mejora en F1 de hasta el 37.53 % con respecto a un detector base.

1. Antecedentes y trabajos relacionados

Los trabajos relacionados que aplican *deep learning* a los datos de teledetección se pueden dividir a grandes rasgos en dos tipos, trabajos *top-down* y *bottom-up*:

- Los trabajos *top-down*, primero construyen un gran conjunto de datos con un importante número de clases de objetos, principalmente objetos que pueden ser reconocidos a partir de imágenes de teledetección, por ejemplo, vehículos o estadios de fútbol. A continuación, los estudios analizan estas imágenes utilizando modelos de clasificación o detección de *deep learning* [CHL17, CHZG14, CZH16, CFWM18, LKM$^+$18, LWC$^+$20, XBD$^+$18, YN10, SWW$^+$21].
- Los trabajos *bottom-up* se centran en la resolución de un problema específico que afecta a una o pocas clases de objetos, por ejemplo, los aeropuertos [BŞH18, CJZ$^+$17, LXZ$^+$19, XZL$^+$18, ZNDX17], árboles [BNLF21, FWC19, GASC$^+$20, SGM$^+$21] y ballenas [GTR$^+$19]. Además, otros trabajos [ZYFL19, CLW$^+$21, CYY$^+$18, LZYF19, HSYF21] se centran en el diseño de nuevos métodos para mejorar la detección, en general, en imágenes de satélite.

Este trabajo pertenece a la segunda categoría, ya que el objetivo final es construir un buen detector de dos infraestructuras críticas específicas, concretamente los aeropuertos y las subestaciones eléctricas. En la sección 1.1 se estudiará este problema. Además, se ofrece un breve resumen de los actuales conjuntos de datos generales que incluyen algunas infraestructuras críticas, los llamados trabajos *top-down* (Sección 1.2) y luego se revisan los enfoques de aprendizaje profundo utilizados en los trabajos *bottom-up* (Sección 1.3).

1.1. Antecedentes

Las infraestructuras críticas son un tipo de uso del suelo por parte del hombre que resulta esencial para el funcionamiento de una sociedad y una economía [OHA$^+$18, XGL$^+$17, YN10]. Cualquier amenaza a estas instalaciones puede causar graves problemas. Algunos ejemplos de infraestructuras críticas son los aeropuertos, las subestaciones eléctricas y los puertos, entre otros. La detección de este tipo de infraestructuras en ortoimágenes de alta resolución es de suma importancia en varios campos como la seguridad, la planificación del uso del suelo y la detección de cambios [CGGGR19, GC17, LQLY19, ZSP$^+$19].

Actualmente, las CNN profundas se han utilizado en gran medida en la clasificación de ortoimágenes de alta resolución [CHL17, CFWM18, YN10] ya que logran buenas precisiones especialmente en la distinción de objetos de escalas similares en imágenes del mismo tamaño y la misma resolución espacial. Sin embargo, la detección de infraestructuras críticas con tamaños y escalas diferentes, por ejemplo, subestaciones eléctricas que cubren una superficie del orden de cientos de m^2 frente a aeropuertos que pueden cubrir hasta cientos de km^2, sigue siendo un reto. Además, a diferencia de los puentes o las autopistas, las infraestructuras como los aeropuertos y las subestaciones eléctricas presentan grandes variaciones de escala intra e interclase. Un ejemplo de esta clase se puede ver en la figura 28. Por otra parte, cada aeropuerto tiene una estructura y una forma completamente diferentes cuando se ve desde el espacio.

La tarea de detección se aborda utilizando datos de teledetección y CNN. Los datos de teledetección consisten en ortoimágenes de alta resolución que pueden obtenerse a partir de vehículos aéreos no tripulados (UAV) (capturados a una altura $< 30km$ y que cubren desde 0.1 hasta $100Km^2$), aviones (a $< 30km$ de altura y que cubren desde 10 hasta $100Km^2$) o satélites ($> 150km$ 10 a 1000 Km^2) [XXZ18]. Obtener grandes cantidades de este tipo de datos es caro. Afortunadamente, algunas fuentes, como Google Earth[1] y Bing

[1]Google Earth: `https://earth.google.com/web`

Figura 28: Imagen de satélite en Guadix (Granada), con una subestación eléctrica como infraestructura crítica.

Maps[2], permiten descargar imágenes aéreas y de satélite de forma gratuita para la comunidad académica. Sin embargo, la mayoría de los conjuntos de datos existentes sobre el uso del suelo están preparados únicamente para el entrenamiento de modelos de clasificación, no incluyen ni anotaciones para el entrenamiento de modelos de detección ni información sobre la escala o el nivel de zoom de las imágenes. Hasta donde se conoce, ninguna de las bases de datos públicas preparadas para el entrenamiento de modelos de detección proporciona imágenes de algunas infraestructuras críticas como las subestaciones eléctricas.

Este trabajo presenta la metodología de dos niveles *Detection for Different Scale Critical Infrastructures* (DetDSCI) en ortoimágenes con *deep learning*. Se reformula el problema de la detección de infraestructuras críticas en ortoimágenes en dos subproblemas, la detección de infraestructuras críticas a pequeña y gran escala. La metodología DetDSCI consta de dos etapas:

- La primera etapa se basa en un modelo de clasificación de la resolución espacial que analiza la imagen de entrada de 2000×2000 píxeles para estimar su nivel de zoom y determinar así el detector que se utilizará en la siguiente etapa.
- La segunda etapa incluye dos detectores expertos, uno de ellos para infraestructuras críticas pequeñas y otro para grandes. Una vez determinado el nivel de zoom de la imagen de entrada por la primera etapa, el detector seleccionado analizará esa imagen de entrada en función de su resolución espacial. Las infraestructuras de escala media pueden ser detectadas por ambos detectores.

Abordar la detección de infraestructuras críticas de escala demasiado pequeña y demasiado grande

[2]Bing Maps: `https://www.bing.com/maps`

en imágenes de teledetección independientemente de la resolución espacial puede ofrecer un mejor rendimiento. Este estudio se centra en dos infraestructuras críticas representativas, a saber, los aeropuertos y las subestaciones eléctricas. Dado que no existen conjuntos de datos de detección públicos que incluyan ambas categorías de infraestructuras críticas, se ha creado cuidadosamente un conjunto de datos especializado, el conjunto de datos de infraestructuras críticas (CI-dataset). El conjunto CI-dataset está organizado en dos subconjuntos, el conjunto de datos de infraestructuras críticas a pequeña escala (CI-SS) con la clase de subestación eléctrica y el conjunto de datos de infraestructuras críticas a gran escala (CI-LS) con la clase de aeropuerto.

Las principales aportaciones de este trabajo pueden resumirse como sigue:

- A diferencia del proceso tradicional adoptado para construir la mayoría de los conjuntos de datos, se ha seguido un proceso dinámico para construir el conjunto CI-dataset de alta calidad organizado en dos escalas, CI-SS para las infraestructuras críticas de pequeña escala y CI-LS para las infraestructuras críticas de gran escala. Este proceso puede utilizarse para incluir más tipos de infraestructuras. El conjunto de datos CI está disponible a través de este enlace[3].
- Se presenta la metodología DetDSCI, *Detection for Different Scale Critical Infrastructures* en ortoimágenes en dos etapas con *deep learning*. La metodología DetDSCI determina primero la resolución espacial de la imagen de entrada y luego la analiza según su resolución espacial utilizando el detector experto apropiado. Esta metodología supera a los detectores de base entrenados en el conjunto de datos de alta calidad. El código de la metodología DetDSCI está disponible a través de este enlace[4].

1.2. Trabajos *top-down*

La mayoría de las bases de datos proporcionadas por los trabajos *top-down* son conjuntos de datos multiclase que incluyen algunas infraestructuras críticas, anotadas para la tarea de clasificación de imágenes, lo que limita su utilidad. Véase el resumen en la tabla III.1 donde sólo unos pocos conjuntos de datos están preparados para la tarea de detección.

Tabla III.1: Características de los conjuntos de datos generales que incluyen algunas infraestructuras críticas.

Dataset	#Clases (#Infraestructura)	#Imágenes (#Instancias)	#Ancho imagen	Fuente	Resolución	Anotación
LULC[YN10]	21 (7)	2100 (2100)	256	National Map	30cm	Clasificación
NWPU RESISC45[CHL17]	45 (13)	31500 (31500)	256	Google Earth	20cm-30cm	Clasificación
fMoW[CFWM18]	62 (25)	523846 (132716)	N/A	OpenStreetMap	31cm-1.6m	Clasificación
NWPU VHR-10[CHZG14]	10 (4)	800 (3651)	∼1000	Google Earth	15cm-12m	BB horizontal
xView[LKM+18]	60 (9)	1400 (1000000)	3000	DigitalGlobe	31cm	BB horizontal
DIOR[LWC+20]	20 (11)	23463 (192472)	800	Google Earth	30cm-50cm	BB horizontal
DOTA[XBD+18]	15 (6)	2806 (188282)	800∼4000	Google Earth	15cm-12m	BB orientado

Por ejemplo, en [YN10], los autores crearon un conjunto de datos LULC organizado en 21 clases. Cada clase contiene 100 imágenes de tamaño 256×256 píxeles. Los autores de [CHL17] proporcionan un conjunto de datos denominado NWPU-RESISC45. Este conjunto de datos está compuesto por 31500

[3]CI-dataset: `https://dasci.es/transferencia/open-data/ci-dataset/`

[4]DetDSCI methodology: `https://github.com/FPerezHernandez92/DetDSCI-Methodology`

imágenes de 256×256 píxeles, en 45 clases con 700 imágenes cada una. NWPU-RESISC45 incluye imágenes con una gran variación en cuanto a traslación, resolución espacial, punto de vista, pose del objeto, iluminación, fondo y oclusión. Además, tiene una gran diversidad dentro de la clase y una gran similitud entre clases. Functional Map of the World (fMoW) [CFWM18] es un conjunto de datos que contiene un total de 523846 imágenes con una resolución espacial de 0.31 y 1.60 metros por píxel. Incluye 62 clases con 132716 instancias de OpenStreetMap. Estos conjuntos de datos están preparados para la tarea de clasificación de imágenes y, por tanto, no son útiles para la tarea de detección.

Ejemplos de conjuntos de datos preparados para la tarea de detección de objetos son NWPU VHR-10, xView, DIOR y DOTA. El conjunto de datos NWPU VHR-10 [CHZG14] está organizado en 10 clases, cada clase contiene 800 imágenes de 1000 píxeles de ancho. Contiene principalmente objetos a pequeña escala como aviones, barcos, tanques de almacenamiento, pistas de béisbol, pistas de tenis, pistas de baloncesto, pistas de atletismo, campos de fútbol, puertos, puentes y vehículos. Los autores de [LKM+18] presentaron el conjunto de datos xView para detectar 60 clases de objetos con más de 1 millón de instancias. Estas clases se centran en vehículos y objetos de pequeña escala y las imágenes tienen un ancho de 3000 píxeles. DIOR, un nuevo conjunto de datos fue publicado en [LWC+20], donde 23463 imágenes y 192472 instancias cubrían 20 clases de objetos. El conjunto de datos DIOR tiene una gran variedad de tamaños de objetos y se centra en la detección con una anchura en las imágenes de 800 píxeles. El conjunto de datos DOTA [XBD+18] se compone de 15 clases de objetos a pequeña escala con 2806 imágenes de Google Earth donde las instancias totales son 188282. El tamaño de las imágenes está entre 800 y 4000 píxeles, y están etiquetadas con cajas delimitadoras orientadas.

Aunque los cuatro últimos conjuntos de datos están preparados para la tarea de detección de objetos, no se centran en ningún problema específico, ya que todos ellos son tipos de objetos visibles desde el espacio. Además, ninguno de estos conjuntos de datos incluye subestaciones eléctricas y sólo DIOR incluye la categoría de aeropuerto.

1.3. Trabajos *bottom-up*

Un gran número de trabajos *bottom-up* se centran en mejorar la detección de aeropuertos. En [ZNDX17], los autores proponen un método que utiliza CNN para la detección de aeropuertos en imágenes de satélite. El método propuesto consta principalmente de tres pasos, la propuesta de regiones, la identificación con CNN y la optimización de la localización. El modelo se probó en un conjunto de datos de imágenes que incluía 170 aeropuertos diferentes y 30 no aeropuertos. Todas las imágenes ópticas de satélite probadas se obtuvieron de Google Earth con una resolución de $8m \times 8m$ y un tamaño de unos 3000×3000 píxeles. El método propuesto en [BŞH18] detecta primero varias regiones en RSI, y luego utiliza estas regiones candidatas para entrenar una arquitectura CNN. Los tamaños de las imágenes de los aeropuertos eran de 3000×2000 píxeles con una resolución de 1m. Se recogieron un total de 92 imágenes. En [CJZ+17], los autores desarrollaron una estrategia de explotación de ejemplos difíciles y de equilibrio de pesos para construir una novedosa CNN de extremo a extremo para la detección de aeropuertos. Diseñaron una capa de extracción de ejemplos difíciles para seleccionar automáticamente los ejemplos difíciles por sus pérdidas e implementaron una nueva función de pérdida de peso equilibrado para optimizar la CNN. Los autores en [XZL+18] propusieron un método de detección de aeropuertos de extremo a extremo basado en CNN. Además, se ha empleado una estrategia de optimización cruzada para conseguir compartir la capa de convolución entre las redes de propuesta de regiones en cascada y las posteriores redes de detección multiumbral, y este enfoque disminuyó significativamente el tiempo de detección. Una vez detectado el aeropuerto, utilizan un detector de aviones para obtener estas instancias. Para abordar la insuficiencia de los modelos tradicionales en la detección de aeropuertos bajo fondos complicados a partir de imágenes

de teledetección, los autores en [LXZ+19] propusieron un modelo de detección y expresión jerárquica de aeropuertos de teledetección de extremo a extremo basado en CNN profundas transferibles.

Varios estudios se centran en mejorar la detección de objetos generales en imágenes de teledetección. Por ejemplo, en [ZYFL19] los autores proporcionaron un conjunto de datos de teledetección llamado HRRSD y diseñaron una CNN llamada HRCNN basada en la técnica de propuesta deformable para mejorar la detección de estas clases. En [CLW+21], los autores aplicaron en primer lugar un módulo de mejora de características de doble atención (DAFE) para destacar selectivamente las características de información de múltiples resoluciones. A continuación, introdujeron un módulo de *Context Feature Enhancement* (CFE) para aprovechar al máximo la abundante información surgida en los objetos de teledetección. Los autores de [CYY+18] propusieron una CNN discriminante (D-CNN) para clasificar escenas de teledetección. Demostraron que la D-CNN mapea las imágenes de la misma escena cerca unas de otras, mientras que las imágenes de diferentes escenas se mapean muy lejos unas de otras. En [LZYF19], los autores presentaron la red GACL para mejorar la detección de objetos a pequeña escala en imágenes de teledetección. El modelo utiliza las características globales para guiar la atención de las características convolucionales locales, y el proceso de predicción centrado en el eje toma el proceso de agrupación de un solo eje para evitar la perturbación de la predicción de coordenadas. Los autores de [HSYF21] desarrollaron una CNN especialmente diseñada para la detección de nubes en imágenes ópticas de teledetección.

2. Metodología DetDSCI

En esta sección se presenta la metodología DetDSCI, cuyo objetivo es abordar la detección de aeropuertos y subestaciones eléctricas de tamaños y formas muy diferentes en grandes áreas representadas por imágenes de satélite, véase la ilustración de la figura 29. Se definen dos amplios rangos de resoluciones espaciales también llamados niveles de zoom. Véase la correspondencia entre el nivel de zoom y la resolución espacial en la tabla III.2. El primer rango incluye los niveles de zoom de [14,17] y el segundo rango incluye los niveles de zoom de [18,23]. Estos intervalos se han seleccionado experimentalmente como se describe en la siguiente sección.

Tabla III.2: Correspondencia entre la resolución espacial y el nivel de zoom.

Infraestructuras críticas grandes		Infraestructuras criticas pequeñas	
Nivel de zoom	Resolución espacial (m^2/píxel)	Nivel de zoom	Resolución espacial (m^2/píxel)
14	6.2	18	0.39
15	3.1	19	0.19
16	1.55	20	0.10
17	0.78	21	0.05
		22	0.02
		23	0.01

Para reducir el número de falsos positivos a causa de las diferencias en los distintos niveles de zoom, la metodología DetDSCI distingue primero entre los dos rangos de niveles de zoom y luego aplica el detector correspondiente según la resolución espacial de cada imagen de entrada. En concreto, DetDSCI es en realidad un proceso de dos etapas, tal y como se ilustra en la figura 30. La primera etapa determina si la imagen de entrada pertenece al primer o segundo intervalo de niveles de zoom. En función del intervalo de

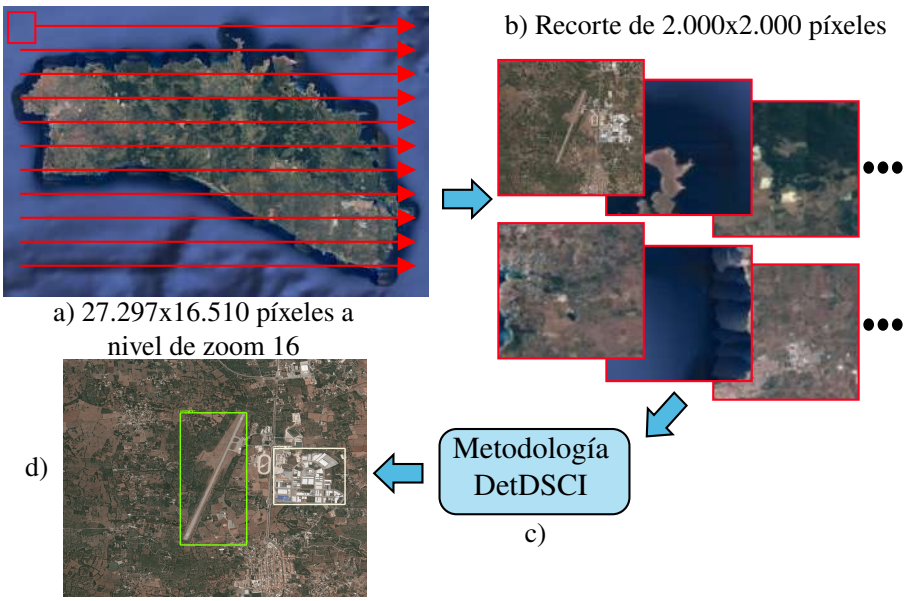


Figura 29: La metodología DetDSCI de detección aplicada a la isla de Menorca (España). (a) Enfoque de procesamiento de ventanas deslizantes. (b) Recortes obtenidos de 2000 × 2000 píxeles. (c) Metodología DetDSCI aplicada a cada recorte. (d) Imagen de salida con los resultados de la detección.

nivel de zoom seleccionado, la segunda etapa analiza esa imagen utilizando el detector especializado en ese grupo específico de infraestructuras críticas.

El siguiente código resume la metodología DetDSCI:

```
DetDSCI(imagen):
    nivel_zoom_imagen = ClassificadorNivelZoom(imagen)
    if nivel_zoom_imagen <= 17:
        clase = DetectorEscalaGrande(imagen)
    elif nivel_zoom_imagen >= 18:
        clase = DetectorEscalaPequenia(imagen)
    return clase
```

2.1. Etapa 1: Estimación de la resolución espacial en la imagen de entrada

Para distinguir entre infraestructuras críticas demasiado grandes y pequeñas, se consideran dos intervalos de niveles de zoom, [14,17] y [18,23]. Las infraestructuras demasiado grandes pueden reconocerse visualmente en las imágenes de 2000 × 2000 píxeles de los niveles de zoom 14, 15, 16 y 17. Véase un ejemplo en la figura 31. Mientras que las infraestructuras a escala demasiado pequeña pueden reconocerse visualmente en imágenes de 2000 × 2000 píxeles de los niveles de zoom 18, 19, 20, 21, 22 y 23. Véase un ejemplo en la figura 32. Las infraestructuras de tamaño medio, como los puentes, pueden incluirse en grupos pequeños y grandes.

En las figuras 31 y 32 se descubre que las subestaciones eléctricas son difíciles de reconocer por el ojo humano en el nivel de zoom 18 y los aeropuertos son difíciles de reconocer en el nivel de zoom 14. Esto se debe a que el número de píxeles proporcionados con estos niveles de zoom no dan suficiente información

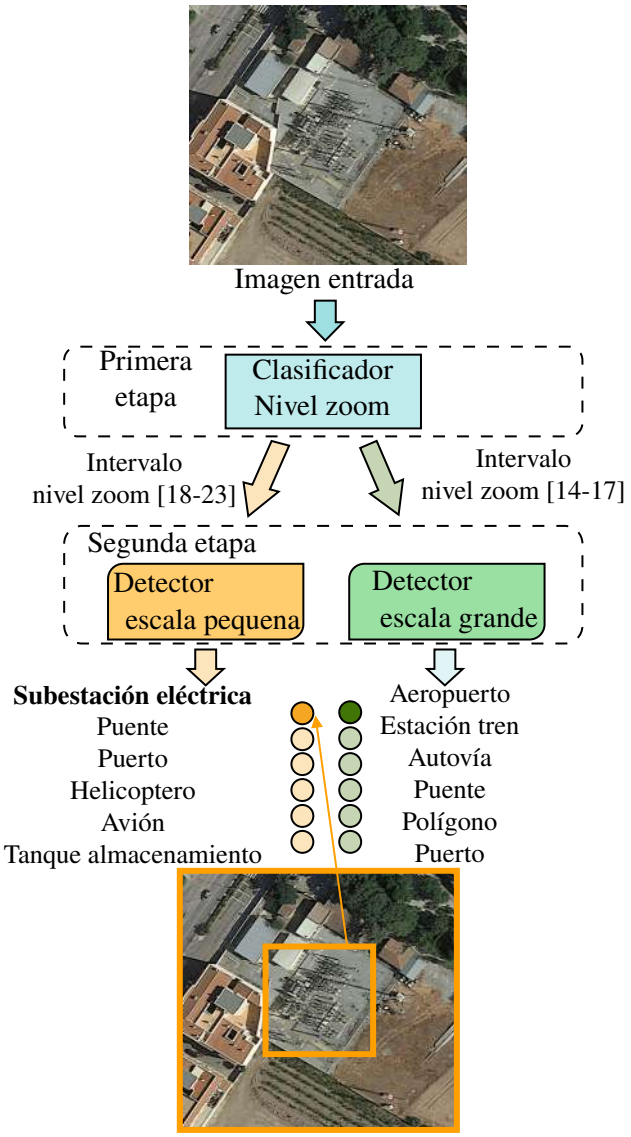
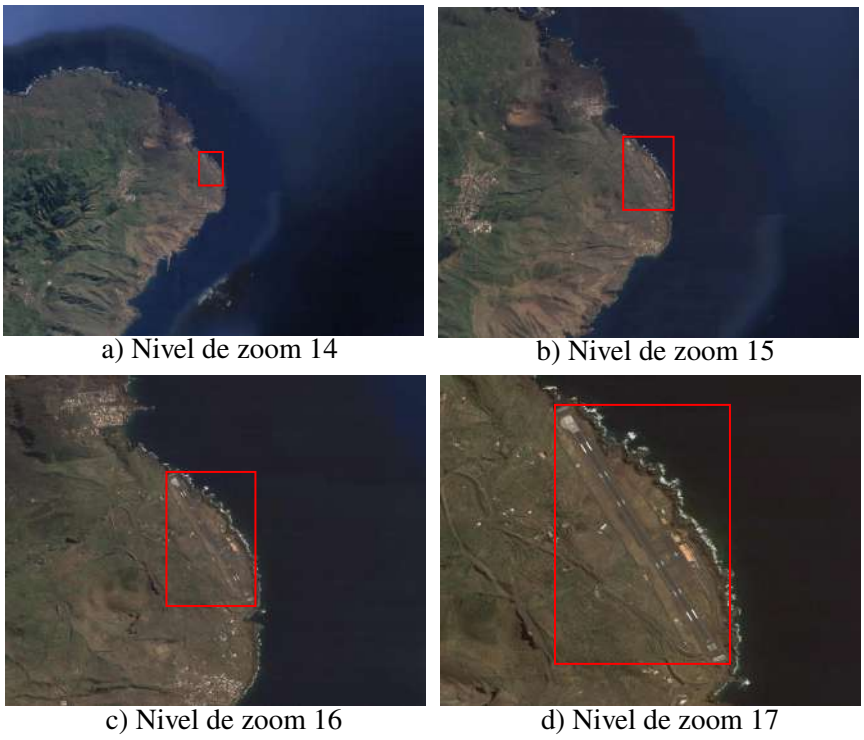


Figura 30: Metodología DetDSCI.

sobre los objetos objetivo. Paralelamente, algunas imágenes de 2000×2000 píxeles no pueden contener todo el aeropuerto en el nivel de zoom 17. Del mismo modo, algunas imágenes de 2000×2000 píxeles no pueden contener toda la subestación eléctrica en el nivel de zoom 23. A pesar de todo, la inclusión de estos niveles de zoom en el conjunto de datos de entrenamiento mejora la solidez del detector, como se verá en la sección 4.2.1.

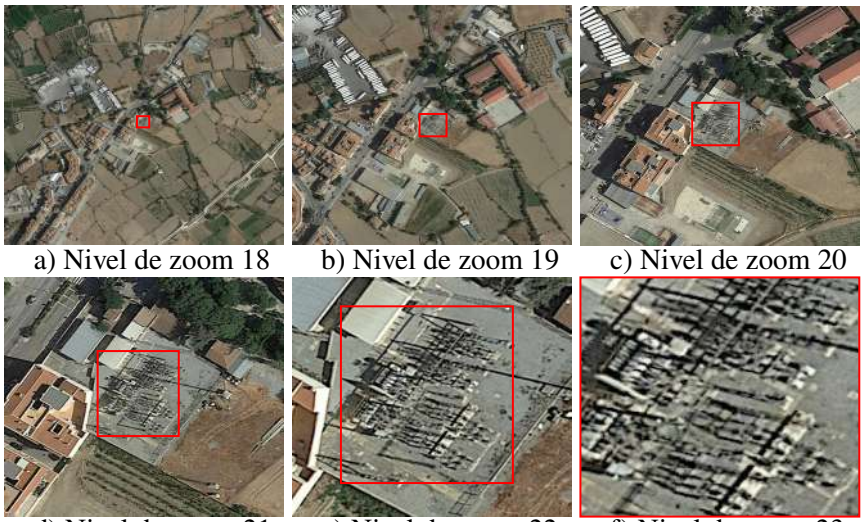
La primera etapa de DetDSCI distingue entre estos dos intervalos, intervalo de niveles de zoom grande [14,17] y pequeño [18,23]. Esta etapa se basa en un modelo de clasificación binaria que analiza la imagen de entrada para determinar su intervalo de nivel de zoom y, por tanto, determina el detector más adecuado que se utilizará en la segunda etapa.



a) Nivel de zoom 14 b) Nivel de zoom 15

c) Nivel de zoom 16 d) Nivel de zoom 17

Figura 31: Cuatro imágenes del aeropuerto de El Hierro (latitud: 27.81402^{o}N, longitud: -17.88518^{o}S, Islas Canarias, España) con niveles de zoom 14(a), 15(b), 16(c) y 17(d), obtenidas de Google Maps.



a) Nivel de zoom 18 b) Nivel de zoom 19 c) Nivel de zoom 20

d) Nivel de zoom 21 e) Nivel de zoom 22 f) Nivel de zoom 23

Figura 32: Seis imágenes de la subestación eléctrica de Guadix (latitud: 37.30853^{o}N, longitud: -3.12997^{o}S, Granada, España) con niveles de zoom 18(a), 19(b), 20(c), 21(d), 22(e) y 23(f), obtenidas de Google Maps.

2.2. Etapa 2: Detección de infraestructuras críticas

El intervalo del nivel de zoom estimado en la primera etapa se utilizará para guiar la selección del detector en la segunda etapa. En concreto, esta etapa se basa en dos modelos de detección:

- El primer modelo de detección se aplica a las infraestructuras a gran escala. Se consideran seis clases de infraestructuras, en concreto, aeropuerto, puente, puerto, polígono industrial, autovía y estación de tren. En la figura 33 se muestran ejemplos de estas clases.
- El segundo modelo de detección se aplica a las infraestructuras de pequeña escala. Se consideran seis clases, en concreto, subestación eléctrica, puente, avión, puerto, tanque de almacenamiento y helicóptero. La figura 34 muestra ejemplos de estas clases.

Cabe mencionar que la inclusión de nuevas clases en ambos detectores se basó en el estudio experimental preliminar que se explica en la siguiente sección.

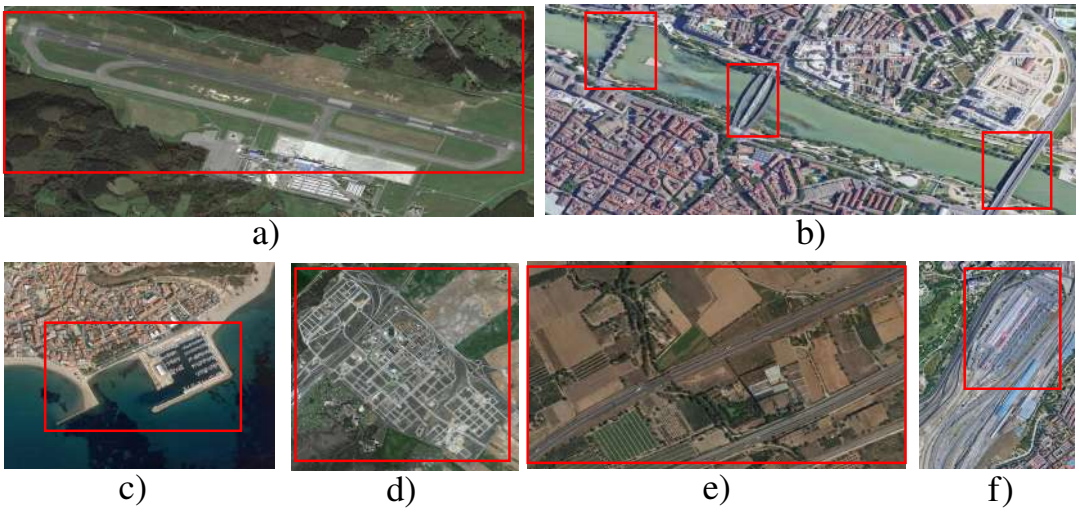


Figura 33: Ejemplos de las clases consideradas por el modelo de detección de grandes infraestructuras, de izquierda a derecha: aeropuerto(a), puentes(b), puerto(c), zona industrial(d), autovía(e) y estación de tren(f).

3. Construcción de la base de datos CI-Dataset guiada por el rendimiento del modelo

Es bien sabido que la construcción de modelos de calidad requiere conjuntos de datos de buena calidad, también llamados datos inteligentes o *Smart Data* [LGGRG+20]. El concepto de datos inteligentes incluye todos los métodos de preprocesamiento que mejoran el valor y la veracidad de los datos. En el contexto de la detección de objetos, normalmente se construyen primero conjuntos de datos de entrenamiento y luego se analizan utilizando modelos de aprendizaje automático. Este procedimiento clásico sólo es adecuado cuando los objetos implicados tienen tamaños similares y pueden identificarse correctamente con la misma resolución espacial.

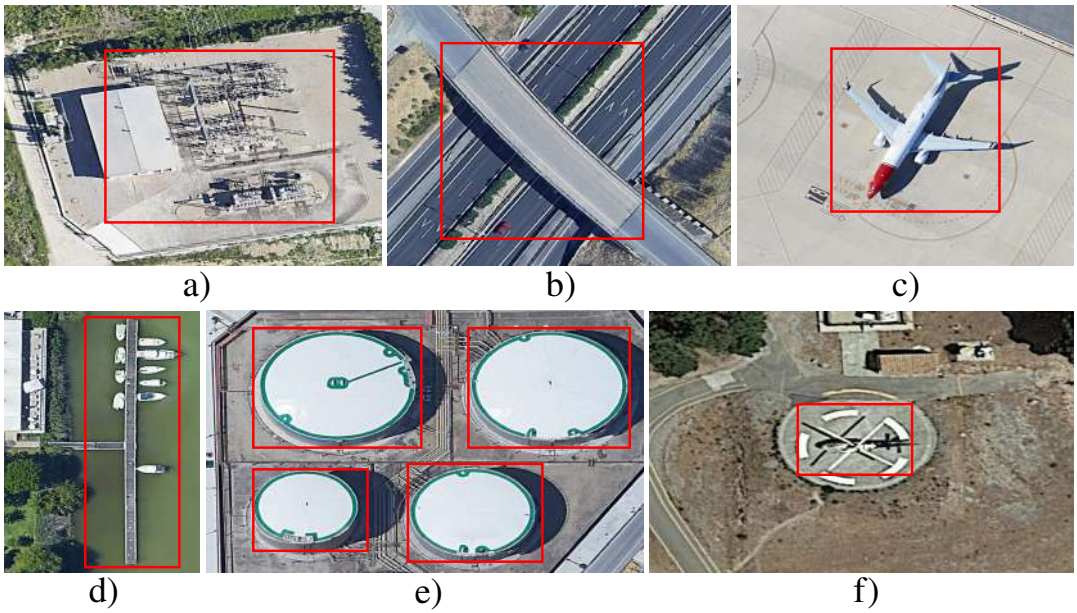


Figura 34: Ejemplos de las clases consideradas en el modelo de detección de pequeñas infraestructuras, de izquierda a derecha: subestación eléctrica(a), puente(b), avión(c), puerto(d), tanques de almacenamiento(e) y helicóptero(f).

Para superar estas limitaciones, se construye el conjunto de datos de infraestructuras críticas, CI-Dataset, guiándose por el rendimiento de uno de los detectores más robustos, Faster R-CNN ResNet 101 V1. Se organiza el conjunto de datos CI en dos subconjuntos, uno para infraestructuras críticas a pequeña escala, CI-SS, y otro para infraestructuras críticas a gran escala, CI-LS. El proceso de construcción de ambos subconjuntos es dinámico y está guiado por el rendimiento del modelo de detección en la clase de subestaciones eléctricas para CI-SS y la clase de aeropuertos para CI-LS. Esta sección describe el proceso de construcción utilizado para obtener el conjunto de datos CI-Dataset final de alta calidad para la detección de subestaciones eléctricas y aeropuertos.

El proceso dinámico guiado por el modelo de detección se basa en tres pasos principales:

- **Paso 1: Construcción del conjunto inicial para la clase objetivo:** En primer lugar, se selecciona la combinación de niveles de zoom en la que los aeropuertos y las subestaciones eléctricas pueden ser reconocidos por el ojo humano. A continuación, se descargan imágenes para cada una de estas dos clases con diferentes niveles de zoom. Después, se selecciona la combinación de niveles de zoom más adecuada guiándose por el rendimiento del modelo de detección.
- **Paso 2: Ampliación del conjunto de datos con más clases:** Se analizan todas las clases de objetos que pueden confundirse con la clase objetivo y, por tanto, pueden causar falsos positivos (FP). Todos estos posibles FP se obtienen de conjuntos de datos públicos y se incluyen en el conjunto CI-Dataset. A continuación, se analiza el rendimiento del modelo para seleccionar las clases de objetos finales que se incluirán.
- **Paso 3: Aumento del tamaño del conjunto de entrenamiento:** Se aumenta el número de instancias de las clases finales en el conjunto de entrenamiento utilizando nuevas imágenes obtenidas de Google Maps.

Para simplificar, se nombran tres versiones diferentes de los conjuntos de datos de entrenamiento, de test y del modelo de detección según el paso de construcción descrito en la tabla III.3. Al final de este proceso, se obtienen los conjuntos de datos finales de entrenamiento y test de CI-Dataset.

Tabla III.3: Nombres de los subconjuntos de entrenamiento y de test de CI-Dataset y del correspondiente modelo de detección creado en cada paso del proceso.

	Entrenamiento	Test	Modelo de detección
Paso 1	CI-SS_train_alpha	CI-SS_test_alpha	CI-SS_Det_alpha
Paso 2	CI-SS_train_beta	CI-SS_test_stable	CI-SS_Det_beta
Paso 3	CI-SS_train_stable	CI-SS_test_stable	CI-SS_Det_stable
Paso 1	CI-LS_train_alpha	CI-LS_test_alpha	CI-LS_Det_alpha
Paso 2	CI-LS_train_beta	CI-LS_test_stable	CI-LS_Det_beta
Paso 3	CI-LS_train_stable	CI-LS_test_stable	CI-LS_Det_stable

3.1. Paso 1: Construcción del conjunto inicial para la clase objetivo

El primer proceso consiste en seleccionar cuidadosamente los niveles de zoom en los que los objetos considerados caben en una imagen de 2000×2000 píxeles y pueden ser reconocidos por el ojo humano. Las ortoimágenes de este tamaño pueden capturar infraestructuras críticas a pequeña escala dentro del intervalo de nivel de zoom 18 a 23 (véase la figura 34) y las infraestructuras críticas a gran escala dentro del intervalo de nivel de zoom 14 a 17 (véase la figura 33). Para construir el conjunto CI-Dataset, se han utilizado dos servicios para visualizar y descargar imágenes de Google Maps, SAS Planet[5] y Google Maps API[6].

Aunque todos los niveles de zoom seleccionados proporcionan información útil para el entrenamiento del modelo de detección, los niveles de zoom más bajos, 14 y 18, y más altos, 17 y 22 y 23, requieren un preprocesamiento manual específico para que se ajusten a 2000×2000 píxeles[7] para que puedan utilizarse en el entrenamiento del modelo de detección. Para el proceso de test, no se aplica ningún preprocesamiento y se descartan los niveles de zoom 14 y 17 para escala grande (figura 35 (a)) y 18, 22 y 23 para las infraestructuras de pequeña escala (figura 35 (b)). Es decir, se consideran los niveles de zoom [19,21] para subestación eléctrica y [15,16] para la clase aeropuerto, en el conjunto de test. Una vez seleccionados los niveles de zoom para el proceso de entrenamiento, se descargan las imágenes de la clase objetivo para construir los subconjuntos CI-SS y CI-LS.

Por último, una vez construido el conjunto de datos de la clase objetivo, se analizan todas las combinaciones de niveles de zoom para determinar cuál mejora el proceso de aprendizaje de los modelos de detección. Guiados por el rendimiento del modelo de detección en la clase objetivo, se descartan los niveles de zoom que no ayudan en el proceso de aprendizaje del detector.

Pequeña escala: El conjunto de datos inicial de CI-SS, CI-SS_train_alpha, se construye utilizando las imágenes de subestaciones eléctricas con niveles de zoom desde el 18 hasta el 23. Se descargaron 550 imágenes con diferentes niveles de zoom, como se muestra en la tabla III.4a. Para construir el conjunto de test, CI-SS_test_alpha, se descargaron 75 imágenes de la clase subestación eléctrica con niveles de zoom del 19 al 21, como se muestra en la tabla III.4b.

[5]SAS Planet: `https://www.sasgis.org/`

[6]Google Maps API: `https://cloud.google.com/maps-platform`

[7]El preprocesamiento incluye la fusión de múltiples imágenes, el recorte de una imagen y/o el cambio de tamaño de la imagen obtenida a 2000×2000 píxeles. Obsérvese que este tamaño corresponde a la capa de entrada del modelo de detección.

Figura 35: Niveles de zoom descartados para el test. a) En escala grande se descarta el nivel de zoom 14 al estar los objetos demasiado alejados y el 17 por ocupar demasiado. b) En escala pequeña se descarta el 18 al estar los objetos demasiado alejados y el 22 y 23 por ocupar demasiado.

Tabla III.4: Número de instancias para la clase subestación eléctrica, a) CI-SS_train_alpha, b) CI-SS_test_alpha.

(a)

Nivel de zoom	Subestación eléctrica
18	103
19	103
20	103
21	103
22	103
23	103
Total	618

(b)

Nivel de zoom	Subestación eléctrica
19	27
20	27
21	27
Total	81

Gran escala: La versión inicial del conjunto de datos CI-LS, CI-LS_train_alpha, se construyó utilizando únicamente imágenes de aeropuertos con niveles de zoom del 14 al 17. Se descargaron 160 imágenes de aeropuertos de España y 80 aeropuertos de Francia, como se muestra en la tabla III.5a. Para construir el conjunto de test inicial, CI-LS_test_alpha, se descargaron 32 imágenes de aeropuertos españoles con dos niveles de zoom, 15 y 16, como se muestra en la tabla III.5b.

Tabla III.5: Número de instancias en la clase aeropuerto, a) CI-LS_train_alpha, b) CI-LS_test_alpha.

(a)

Nivel de zoom	Aeropuerto
14	60
15	69
16	251
17	124
Total	504

(b)

Nivel de zoom	Aeropuerto
15	17
16	16
Total	33

3.2. Paso 2: Ampliación del conjunto de datos con más clases

Tras un cuidadoso análisis de los FP cometidos por el modelo de detección cuando fue entrenado en el conjunto de datos inicial, se determinaron todas las clases de objetos potenciales que hacen que el detector confunda la clase objetivo con otros objetos diferentes. En esta fase, se analizó el impacto de cada una de estos potenciales FP en el aprendizaje del detector y se amplió el conjunto de datos con más clases de objetos procedentes de conjuntos de datos y fuentes públicas. Si el rendimiento mejora, esa clase de FP potencial se mantiene en el conjunto de datos, en caso contrario, se elimina del conjunto de datos.

Para las infraestructuras de pequeña escala, se añadirá el conjunto de datos DOTA, ya que sus objetos son de escalas similares. Para las infraestructuras a gran escala, se utilizará el conjunto de datos DIOR, ya que contiene infraestructuras de tamaños similares. Para los conjuntos de datos de pequeña y gran escala, también se incluye un gran número de imágenes de las mismas clases descargadas de Google Maps y anotadas manualmente para su detección.

Pequeña escala: Se incluyeron en CI-SS_train_beta todas las clases DOTA enumeradas en la tabla III.6, además de un gran número de imágenes descargadas de Google Maps. A continuación, se eliminó cada clase de DOTA, una por una, y se evaluó su impacto en el rendimiento del detector.

Además, como se ha descubierto que las nuevas clases más relevantes son el puente, puerto, tanque de almacenamiento, avión y helicóptero, el detector se entrena para discriminar también estas clases. Para construir CI-SS_test_stable, se incluyeron 132 imágenes de las cinco nuevas clases, como se resume en la tabla III.7.

Gran escala: Después de analizar los FP con el modelo de detección, se incluyeron tres clases de objetos del conjunto de datos DIOR junto con un gran número de imágenes de las mismas clases descargadas de Google Maps en CI-LS_train_beta, concretamente estación de tren, puente y puerto, y se construyó la clase autovía y polígono industrial, obteniendo el conjunto que aparece en la tabla III.8. Se construyó el conjunto de test, CI-LS_test_stable, incluyendo 114 nuevas imágenes de las cinco clases, como puede verse en la tabla III.9.

3.3. Paso 3: Aumento del tamaño del conjunto de entrenamiento

En esta etapa, se incrementa el número de imágenes de todas las nuevas clases añadidas a ambos subconjuntos de entrenamiento utilizando nuevas imágenes de Google Maps.

Pequeña escala: Como el modelo entrenado por CI-SS_Det_beta confunde la subestación eléctrica

Tabla III.6: Número de instancias en las infraestructuras críticas de pequeña escala, CI-SS_train_beta.

	Google Maps Nivel de zoom						DOTA	Total
	18	19	20	21	22	23		
Subestación eléctrica	103	103	103	103	103	103	-	618
Vehículo largo	0	3	26	5	3	0	16923	16960
Piscina	111	104	62	11	2	0	1732	2022
Helicóptero	0	0	0	0	0	0	630	630
Puente	19	18	5	0	0	0	2041	2083
Avión	0	0	0	0	0	0	7944	7944
Barco	0	0	0	0	0	0	28033	28033
Campo fútbol	4	4	1	0	0	0	311	320
Pista baloncesto	0	0	0	0	0	0	509	509
Pista atletismo	0	0	0	0	0	0	307	307
Vehículo pequeño	0	0	141	234	68	5	26099	26547
Puerto	0	0	0	0	1	0	5937	5938
Pista béisbol	0	0	0	0	0	0	412	412
Pista tenis	6	6	1	0	0	0	2325	2338
Rotonda	25	26	13	1	0	0	385	450
Tanque almacenamiento	23	39	36	12	0	0	5024	5134

Tabla III.7: Número de instancias en la versión final de test de infraestructuras críticas a pequeña escala, CI-SS_test_stable dataset.

Nivel de zoom	Subestación eléctrica	Helicóptero	Puente	Avión	Puerto	Tanque almacenamiento
19	27	8	21	68	57	136
20	27	8	15	35	27	50
21	27	6	13	17	12	24
Total	81	22	49	120	96	210

Tabla III.8: Número de instancias en el conjunto de entrenamiento de infraestructuras críticas a gran escala, CI-LS_train_beta.

		Aeropuerto	Estación tren	Autovía	Puente	Polígono	Puerto
Google	14	60	1	566	1	11	1
Maps	15	69	2	819	1	14	1
nivel de	16	251	2	3207	8	34	1
zoom	17	124	19	2859	4	50	1
DIOR		1327	1011	-	3967	-	5509
Total		1831	1035	7451	3981	109	5513

Tabla III.9: Número de instancias en la versión final de test de infraestructuras críticas a gran escala, CI-LS_test_stable.

Nivel de zoom	Aeropuerto	Estación tren	Autovía	Puente	Polígono	Puerto
15	17	25	518	115	59	32
16	16	22	303	55	27	20
Total	33	47	821	170	86	52

con varios elementos de las zonas urbanas, se incluyeron las zonas urbanas como contexto en las nuevas imágenes de entrenamiento en el resto de las clases. En concreto, se descargaron un total de 1173 nuevas imágenes. Las características de CI-SS_train_stable obtenido se muestran en la tabla III.10.

Tabla III.10: Número de instancias en el conjunto de datos final de entrenamiento de infraestructuras críticas a pequeña escala, CI-SS_train_stable.

	Google Maps Nivel de zoom						DOTA	Total
	18	19	20	21	22	23		
Subestación eléctrica	103	278	267	247	103	103	-	1101
Piscina	111	911	370	141	2	0	1732	3267
Helicóptero	0	20	17	17	0	0	630	684
Puente	19	88	39	19	0	0	2041	2206
Avión	0	13	8	2	0	0	7944	7967
Campo fútbol	4	146	65	40	0	0	311	566
Pista baloncesto	0	91	49	35	0	0	509	684
Pista atletismo	0	4	0	0	0	0	307	311
Puerto	0	1	0	0	1	0	5937	5939
Pista béisbol	0	2	0	0	0	0	412	414
Pista tenis	6	126	46	27	0	0	2325	2530
Rotonda	25	103	38	8	0	0	385	559
Tanque almacenamiento	23	538	249	73	0	0	5024	5907

Gran escala: Se aumentó el tamaño del conjunto de datos CI-LS_train_beta incluyendo 768 nuevas imágenes. Las características de CI-LS_train_estable obtenido se muestran en la tabla III.11.

4. Estudio experimental

En esta sección se presenta todo el análisis experimental que se realizó para obtener el CI-Dataset y la evaluación de la metodología DetDSCI. La sección 4.1 resume la configuración experimental para el análisis. La sección 4.2 proporciona todos los resultados del modelo de detección obtenidos durante el proceso de construcción del CI-Dataset. Por último, la sección 4.3 proporciona el análisis y la comparación de la metodología DetDSCI propuesta.

Tabla III.11: Número de instancias en el conjunto de datos final de entrenamiento de infraestructuras críticas a gran escala, CI-LS_train_stable.

		Aeropuerto	Estación tren	Autovía	Puente	Polígono	Puerto
Google	14	60	5	1012	37	69	17
Maps	15	69	6	1280	37	71	17
nivel de	16	251	6	3947	57	116	27
zoom	17	124	27	4805	168	291	23
DIOR		1327	1011	-	3967	-	5509
Total		1831	1055	11044	4266	547	5593

4.1. Configuración experimental

La construcción dinámica del conjunto de datos requiere el uso de un buen modelo de detección. Después de un cuidadoso análisis experimental, se encontró que Faster R-CNN es el más adecuado para este estudio, ya que logra una buena relación de equilibrio entre velocidad y precisión [HRS+17].

Para el entrenamiento de los modelos de detección, las imágenes fueron redimensionadas a una imagen de 2000×2000 que representa el tamaño requerido de la capa de entrada de los detectores modernos. Es necesario seleccionar cuidadosamente el nivel de zoom para que todo el objeto pueda caber en la imagen.

En los experimentos realizados en las siguientes secciones, se utilizó Keras [C+15] como framework de *deep learning* para la clasificación y TensorFlow [ABC+16] como framework de *deep learning* para la detección.

Para evaluar y comparar el rendimiento se utilizarán estas métricas: *Precisión*, *Recall* y *F1*(ecuación III.1).

$$\begin{aligned} Precision &= \frac{TP}{TP+FP} \\ Recall &= \frac{TP}{TP+FN} \\ F1 &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \end{aligned} \tag{III.1}$$

donde se calcula el número de verdaderos positivos (TP), falsos positivos (FP) y falsos negativos (FN) para cada clase.

El rendimiento de la detección se evalúa en términos de mAP (ecuación III.2) y mAR (ecuación III.3), métricas estándar para tareas de detección de objetos [LMB+14] dadas 100 regiones de salida.

$$mAP = \frac{\sum_{i=1}^{K} AP_i}{K} \qquad AP_i = \frac{1}{10} \sum_{r \in [0,5,\ldots,0,95]} \int_0^1 p(r)dr \tag{III.2}$$

$$mAR = \frac{\sum_{i=1}^{K} AR_i}{K} \qquad AR_i = 2\int_{0,5}^{1} recall(o)do \tag{III.3}$$

donde dadas K categorías de elementos, p representa la precisión y r $recall$ define el área bajo la curva interpolada de precisión-recall para cada clase i. Mientras que o es IoU (intersección sobre la unión) y recall(o) es el correspondiente recall bajo la curva recall-IoU para cada clase i.

El rendimiento de los modelos de detección puede mejorarse con el uso de varias técnicas de optimización, concretamente *data augmentation* (DA) y el análisis de diferentes extractores de características (FE). Las ocho técnicas de DA utilizadas para esta tarea se enumeran en la tabla III.12 y se estudiará su impacto en el rendimiento de cada detector.

Tabla III.12: Técnicas de *data augmentation* por modelo.

Nombre modelo	Técnica de *data augmentation*
DA1	*Normalize image*
DA2	*Random image scale*
DA3	*Random rgb to gray*
DA4	*Random adjust brightness*
DA5	*Random adjust contrast*
DA6	*Random adjust hue*
DA7	*Random adjust saturation*
DA8	*Random distort colour*

Además, se consideran seis extractores de características (FE) enumerados en la tabla III.13 y se entrenan los modelos con o sin las mejores técnicas de DA. Se analizará el impacto de todos estos factores en el rendimiento de cada modelo de detección.

Tabla III.13: Configuración de los extractores de características para diferentes modelos.

Nombre modelo	*Region proposal*	Modelo ResNet	con DA
FE1	Faster R-CNN	ResNet 101 V1	No
FE2	Faster R-CNN	ResNet 101 V1	Sí
FE3	Faster R-CNN	ResNet 152 V1	No
FE4	Faster R-CNN	ResNet 152 V1	Sí
FE5	Faster R-CNN	Inception ResNet V2	No
FE6	Faster R-CNN	Inception ResNet V2	Sí

4.2. Estudio experimental para la construcción de CI-Dataset

En la sección 3 se describe detalladamente el proceso de construcción de CI-Dataset. En esta subsección se presentan los resultados experimentales del modelo de detección en cada etapa de dicho proceso. El rendimiento obtenido en las etapas 1, 2 y 3 se analiza respectivamente en las secciones 4.2.1, 4.2.2 y 4.2.3. Por último, el análisis experimental del uso de las técnicas de DA y de los diferentes FE se proporciona en la sección 4.2.4.

4.2.1. Análisis del paso 1: Construcción del conjunto inicial para la clase objetivo

Una vez construido el conjunto CI-Dataset de la clase objetivo, se analizan todas las combinaciones de niveles de zoom para determinar cuál de ellas mejora el proceso de aprendizaje de los modelos de detección. Aunque el número inicial de imágenes de entrenamiento no es demasiado grande, los modelos aprenden correctamente a distinguir entre las distintas clases. Guiados por el rendimiento del modelo de detección en la clase objetivo, se descartan los niveles de zoom que no ayudan en el proceso de aprendizaje del detector.

Pequeña escala: El rendimiento del primer detector, CI-SS_Det_alpha, entrenado en diferentes combinaciones de niveles de zoom muestra resultados similares como se puede ver en la tabla III.14. Se selecciona la combinación que proporciona el mayor número de imágenes, que es la que incluye todos los niveles de zoom, 18, 19, 20, 21, 22 y 23.

Tabla III.14: Rendimiento (%) de CI-SS_Det_alpha cuando se entrena en diferentes combinaciones de niveles de zoom de CI-SS_train_alpha y testeado en CI-SS_test_alpha.

Combinación nivel de zoom	Precisión	Recall	F1	mAP 0.5 subestación eléctrica	mAP 0.5-0.95 media	mAR 0.5-0.95 media
18,19,20,21,22,23	**96.49**	67.90	79.71	87.45	48.30	60.70
19,20,21,22,23	93.44	70.37	**80.28**	86.23	**51.70**	60.40
18,19,20,21,22	91.94	70.37	79.72	**89.90**	48.70	59.00
20,21,22,23	92.31	59.26	72.18	79.35	43.50	55.80
19,20,21,22	89.39	**72.84**	80.27	89.18	51.60	**62.60**
21,22,23	82.76	29.63	43.64	57.90	28.10	38.40
20,21,22	89.29	61.73	72.99	80.55	44.50	54.40
21,22	82.35	17.28	28.57	51.11	24.50	34.70

Gran escala: El rendimiento del modelo de detección, CI-LS_Det_alpha, en diferentes combinaciones de niveles de zoom muestra que los mejores y resultados más estables se obtienen con la combinación de estos niveles de zoom, 14, 15, 16 y 17, como se puede ver en la tabla III.15.

Tabla III.15: Rendimiento (%) de CI-LS_Det_alpha cuando se entrena en diferentes combinaciones de niveles de zoom en CI-LS_train_alpha y testeado en CI-LS_test_alpha.

Combinación nivel de zoom	Precisión	Recall	F1	mAP 0.5 aeropuerto	mAP 0.5-0.95 media	mAR 0.5-0.95 media
14,15,16,17	**87.76**	**86.00**	**86.87**	**89.52**	**61.30**	**69.10**
14,15,16	78.85	82.00	80.39	84.67	55.50	62.10
15,16,17	68.42	78.00	72.90	87.89	54.50	64.20
15,16	87.23	82.00	84.54	82.66	51.00	57.90

4.2.2. Análisis del paso 2: Extensión del dataset ampliando el número de clases

Una vez ampliado el conjunto CI-Dataset con nuevas clases procedentes de conjuntos de datos públicos, se analizó si las nuevas clases mejoran el rendimiento de los modelos de detección.

Pequeña escala: En primer lugar, se entrenó el modelo con todas las clases de DOTA y la clase de subestación eléctrica construida. A continuación, se analizó el impacto de cada clase de DOTA en el modelo de detección eliminando clase por clase del conjunto de datos de entrenamiento. Como se puede ver en la tabla III.16, la eliminación de las tres clases de DOTA, vehículo pequeño, vehículo grande y barco, mejora el F1 del modelo de detección CI-SS_Det_beta. Esto se debe a que las imágenes de estos objetos proporcionan muy poca información sobre sus características, es decir, están representadas con muy pocos píxeles.

Por lo tanto, el conjunto de datos final CI-SS_train_stable contiene 13 clases, pista de tenis, pista de béisbol, pista de atletismo, pista de baloncesto, campo de fútbol, rotonda y piscina, además de puente, puerto, tanque de almacenamiento, helicóptero, avión y subestación eléctrica.

Tabla III.16: Impacto de la eliminación de cada clase individual de DOTA del CI-SS_train_beta en el rendimiento de la detección (%).

Clases eliminadas	Precisión	Recall	F1
Ninguna	88.28	58.38	70.22
- Vehículo pequeño	**92.61**	59.64	72.53
- Vehículo largo	90.30	62.44	73.81
- Barco	90.67	**67.53**	77.35
- Pista tenis	88.09	63.00	73.39
- Pista béisbol	89.97	66.33	76.31
- Pista atletismo	87.02	65.77	74.84
- Pista baloncesto	91.19	63.80	74.99
- Campo fútbol	93.47	66.64	**77.74**
- Rotonda	90.48	65.28	75.70
- Piscina	90.74	66.55	76.73

Gran escala: Los resultados del modelo de detección, CI-LS_Det_beta, entrenado sobre CI-LS_train_beta, se muestran en la tabla III.17. Como se puede observar en esta tabla, la inclusión de algunas clases del conjunto de datos DIOR aumenta el mAP del modelo de detección en la clase aeropuerto hasta el 85.73 %.

Tabla III.17: Rendimiento (%) de CI-LS_Det_beta cuando se entrena en CI-LS_train_beta y se testea en CI-LS_test_stable.

		CI-LS_Det_beta
mAP 0.5	Media	22.03
	Aeropuerto	**85.73**
	Estación tren	6.98
	Autovía	4.30
	Puente	31.97
	Polígono	2.87
	Puerto	0.31
mAP 0.5-0.95	Media	12.20
	Pequeño	2.00
	Mediano	4.70
	Grande	14.40
mAR 0.5-0.95		22.10

4.2.3. Análisis del paso 3: Aumento del tamaño para el conjunto de entrenamiento

Una vez determinadas las clases finales, se incluyen nuevas imágenes para mejorar el rendimiento de los modelos.

Pequeña escala: En la tabla III.18 se muestra una comparación entre CI-SS_Det_beta y el nuevo CI-SS_Det_stable, entrenado en el CI-SS_train_stable (tabla III.10), y testado en CI-SS_test_stable (tabla III.7). El rendimiento de CI-SS_Det_alpha entrenado y testado sólo en la clase subestación eléctrica se incluye también en la tabla como referencia. Estos resultados muestran claramente que el rendimiento de CI-SS_Det_stable mejora al aumentar el tamaño del conjunto de datos de entrenamiento.

Tabla III.18: Rendimiento (%) de CI-SS_Det_stable y CI-SS_Det_beta en CI-SS_test_stable y CI-SS_Det_alpha.

		CI-SS_Det_alpha (solo sub. ele.)	CI-SS_Det_beta (seis clases)	**CI-SS_Det_stable (seis clases)**
mAP 0.5	Media	**87.45**	54.21	65.98
	Subestación eléctrica	**87.45**	78.88	85.00
	Avión	0.00	82.94	**85.30**
	Helicóptero	0.00	**33.83**	10.39
	Puente	0.00	18.33	**63.16**
	Tanque de almacenamiento	0.00	83.07	**92.28**
	Puerto	0.00	58.66	**59.75**
mAP 0.5-0.95	Media	**48.30**	32.30	38.60
	Pequeño	0.00	15.30	**25.90**
	Mediano	**31.80**	23.50	27.90
	Grande	**49.70**	36.80	43.40
mAR 0.5-0.95		**60.70**	47.80	53.10

Para un análisis más profundo, se observan los TP, FP, FN, Precisión, Recall y F1 como se muestra en la tabla III.19. Como se puede observar, CI-SS_Det_stable reduce sustancialmente el número de FP y logra el mejor valor de F1. Por lo tanto, el modelo CI-SS_Det_stable se utilizará en el resto del trabajo, ya que proporciona el mayor rendimiento en la clase objetivo, la subestación eléctrica.

Tabla III.19: TP, FP, FN, Recall (%), Precisión (%) y F1 (%) en CI-SS_test_stable. CI-SS_Det_stable es entrenado en CI-SS_train_stable y CI-SS_Det_beta es entrenado en CI-SS_train_beta. Por motivos de comparación se añade CI-SS_Det_alpha entrenado solo en subestaciones eléctricas.

	TP	FP	FN	Precisión	Recall	F1
CI-SS_Det_alpha (solo sub. ele.)	117	449	7	20.67	**94.35**	33.91
CI-SS_Det_beta (seis clases)	75	124	49	37.69	60.48	46.44
CI-SS_Det_stable (seis clases)	112	62	12	**64.37**	90.32	**75.17**

Gran escala: Una comparación entre CI-LS_Det_beta y el nuevo CI-LS_Det_stable, entrenado en

CI-LS_train_stable (tabla III.11), testeado en el conjunto de datos CI-LS_test_stable (tabla III.9), se muestra en la tabla III.20. El mAP de CI-LS_Det_alpha entrenado y testeado sólo en la clase aeropuerto se incluye también en la tabla como referencia. Como se puede ver en estos resultados, CI-LS_Det_stable muestra un mAP muy similar en los aeropuertos que CI-LS_Det_beta, pero un mAP mucho mejor en el resto de FP potenciales.

Tabla III.20: Rendimiento (%) de CI-LS_Det_stable y CI-LS_Det_beta testeada en CI-LS_test_stable y CI-LS_Det_alpha.

		CI-LS_Det_alpha (solo aeropuertos)	CI-LS_Det_beta (seis clases)	**CI-LS_Det_stable (seis clases)**
mAP 0.5	Media	**89.52**	22.03	36.48
	Aeropuerto	**89.52**	85.73	85.37
	Estación tren	0.00	6.98	**26.45**
	Autovía	0.00	4.30	**5.16**
	Puente	0.00	31.97	**40.53**
	Polígono	0.00	2.87	**20.96**
	Puerto	0.00	0.31	**40.40**
mAP 0.5-0.95	Media	**61.30**	12.20	18.80
	Pequeño	0.00	2.00	**2.40**
	Mediano	0.00	4.70	**6.50**
	Grande	**61.30**	14.40	23.00
mAR 0.5-0.95		**69.10**	22.10	33.90

Una comparación con CI-LS_Det_stable entrenado en CI-LS_train_stable y testeado en CI-LS_test_stable se proporciona en la tabla III.21. En general, CI-LS_Det_table proporciona el F1 más alto.

Tabla III.21: Comparación de TP, FP, FN, TN, Precisión (%), Recall (%) y F1 (%) de CI-LS_Det_stable entrenado en CI-LS_train_stable y testeado en CI-LS_test_stable con CI-LS_Det_beta y CI-LS_Det_alpha. CI-LS_Det_alpha se entrena y se testea sólo en la clase aeropuerto.

	TP	FP	FN	Precisión	Recall	F1
CI-LS_Det_alpha (solo aeropuertos)	29	19	1184	60.42	2.39	4.60
CI-LS_Det_beta (seis clases)	236	35	977	87.08	19.46	31.81
CI-LS_Det_stable (seis clases)	334	39	879	**89.54**	**27.54**	**42.12**

4.2.4. Análisis de la mejora de los modelos de detección

La selección de las técnicas de DA y FE adecuadas puede mejorar sin duda el rendimiento del modelo de detección. Se consideran ocho técnicas de DA enumeradas en la tabla III.12 y se estudia su impacto en el rendimiento de cada detector. Además, se consideran seis FE enumerados en la tabla III.13 y se entrenan los modelos con o sin las mejores técnicas DA. Se analiza el impacto de todos estos factores en el rendimiento de cada modelo de detección.

Pequeña escala: La tabla III.22 muestra el rendimiento de CI-SS_Det_stable al aplicar individualmente diferentes técnicas DA sobre CI-SS_train_stable. Como se puede observar en esta tabla, aplicando DA8, *random distort colour*, se consiguen los mejores resultados en este modelo.

Tabla III.22: Resultados (%) de los diferentes modelos con técnicas DA en CI-SS_train_stable y CI-SS_test_stable.

		DA1	DA2	DA3	DA4	DA5	DA6	DA7	**DA8**
mAP 0.5	Media	22.26	67.85	66.84	68.07	66.45	64.83	64.67	**69.07**
	Subestación eléctrica	0.01	**84.89**	83.65	83.36	82.35	83.23	82.81	82.30
	Avión	41.34	83.23	**88.72**	88.08	82.35	88.06	85.69	86.70
	Helicóptero	0.02	19.82	16.48	14.39	14.99	12.42	10.32	**24.52**
	Puente	15.83	64.90	61.18	**65.86**	62.84	55.08	60.38	64.96
	Tanque almacenamiento	64.28	90.25	89.44	**91.66**	91.16	91.29	91.47	89.88
	Puerto	12.11	64.02	61.55	65.05	65.03	58.79	57.32	**66.07**
mAP 0.5-0.95	Media	12.80	38.70	39.20	39.30	39.20	38.80	38.40	**39.50**
	Pequeño	0.00	23.30	14.10	24.40	23.80	21.80	**31.00**	13.50
	Mediano	2.60	26.50	25.60	27.50	**28.70**	28.20	26.20	26.60
	Grande	18.90	43.70	44.90	44.70	44.30	43.60	43.70	**45.60**
mAR 0.5-0.95		23.50	54.20	54.40	53.50	**54.70**	54.10	52.80	54.20

La tabla III.23 muestra el impacto de las diferentes FE y DA en el rendimiento de CI-SS_Det_stable. Como puede verse, el mejor mAP se obtiene cuando se utiliza Faster R-CNN ResNet 101 V1 con las técnicas FE2 y DA. Este modelo de detección será el nuevo CI-SS_Det_stable.

Tabla III.23: Resultados (%) de diferentes técnicas de FE con o sin DA en CI-SS_train_stable y CI-SS_test_stable.

		FE1	**FE2**	FE3	FE4	FE5	FE6
mAP 0.5	Media	65.98	**68.97**	63.16	65.39	65.83	63.96
	Subestación eléctrica	85.00	85.19	83.05	87.55	82.73	**87.78**
	Avión	85.30	84.43	85.81	80.91	**86.29**	84.96
	Helicóptero	10.39	23.14	6.83	12.48	**48.03**	6.23
	Puente	**63.16**	62.38	48.45	50.31	60.54	39.71
	Tanque almacenamiento	**92.28**	88.97	91.01	90.89	90.93	91.82
	Puerto	59.75	69.70	63.82	70.22	69.71	**73.29**
mAP 0.5-0.95	Media	38.60	**40.20**	36.70	37.60	36.50	37.60
	Pequeño	**25.90**	13.30	4.70	3.10	2.70	3.90
	Mediano	27.90	**29.90**	23.60	21.50	29.70	28.60
	Grande	43.40	**46.30**	42.20	44.50	40.70	42.10
mAR 0.5-0.95		53.10	**54.10**	51.20	53.10	50.70	51.30

Gran escala: La tabla III.24 muestra el rendimiento de CI-LS_Det_stable al aplicar diferentes técnicas DA sobre CI-LS_train_stable. Estos resultados muestran que aplicando DA3, *random rgb to gray*,

se consiguen los mejores resultados de detección.

Tabla III.24: Resultados (%) de los diferentes modelos con técnicas DA en CI-LS_train_stable y CI-LS_test_stable.

		DA1	DA2	**DA3**	DA4	DA5	DA6	DA7	DA8
mAP 0.5	Media	3.61	35.91	**37.11**	36.98	36.62	35.04	36.34	36.98
	Aeropuerto	19.54	85.71	90.31	85.75	90.87	**91.50**	88.18	85.84
	Estación tren	0.07	20.72	**27.98**	26.12	23.53	15.84	19.50	23.39
	Autovía	0.36	4.89	6.19	5.92	6.36	5.20	5.81	**6.63**
	Puente	0.35	39.44	37.78	40.44	36.33	35.92	36.35	**45.05**
	Polígono	0.11	17.05	21.02	21.05	15.85	15.53	**22.06**	15.04
	Puerto	1.22	**47.64**	39.37	42.62	46.76	46.24	46.13	45.94
mAP 0.5-0.95	Media	1.60	18.50	**19.30**	18.20	18.30	18.50	17.90	17.70
	Pequeño	0.10	3.40	3.00	**7.00**	2.20	3.50	2.30	5.20
	Mediano	0.00	6.20	**7.30**	6.60	6.30	6.70	6.30	6.00
	Grande	3.00	20.70	22.40	21.10	21.70	20.80	21.50	**23.00**
mAR 0.5-0.95		13.10	34.80	34.50	**35.40**	33.40	34.20	34.50	34.70

La tabla III.25 muestra el impacto de las diferentes FE y DA en CI-LS_Det_stable. Como se puede ver el mejor rendimiento se obtiene con Faster R-CNN Inception ResNet V2 con FE5 y sin técnicas DA. Este modelo será el nuevo CI-LS_Det_stable en el resto de la experimentación.

Tabla III.25: Resultados (%) de las diferentes técnicas de FE con o sin DA en CI-LS_train_stable y CI-LS_test_stable.

		FE1	FE2	FE3	FE4	**FE5**	FE6
mAP 0.5	Media	36.48	37.52	37.67	38.05	**42.34**	40.98
	Aeropuerto	85.37	86.46	84.03	**87.70**	86.01	87.21
	Estación tren	26.45	24.17	**34.20**	22.31	27.76	22.43
	Autovía	5.16	5.53	4.80	5.77	5.95	**8.01**
	Puente	40.53	47.81	36.69	48.86	**57.27**	54.25
	Polígono	20.96	17.43	23.53	17.54	**23.64**	22.38
	Puerto	40.40	43.71	42.78	46.13	**53.41**	51.63
mAP 0.5-0.95	Media	18.80	18.30	18.80	18.50	**20.30**	20.10
	Pequeño	2.40	5.70	3.20	6.50	**9.70**	7.70
	Mediano	6.50	7.30	6.30	6.70	**8.50**	7.20
	Grande	**23.00**	21.60	22.00	22.90	22.50	22.40
mAR 0.5-0.95		33.90	36.30	35.10	35.20	35.20	**37.70**

4.3. Estudio experimental de la metodología DetDSCI

Una vez construido el conjunto de datos CI-Dataset y entrenados los modelos finales en las infraestructuras críticas de pequeña y gran escala, se desarrolla el clasificador de nivel de zoom para la metodología

DetDSCI. La construcción del clasificador de nivel de zoom se presenta en la sección 4.3.1 y el análisis de la metodología DetDSCI se muestra en la sección 4.3.2.

4.3.1. Construcción del clasificador por niveles de zoom

En la primera etapa de la metodología DetDSCI, un clasificador de niveles de zoom analiza la imagen de entrada y determina la escala de esta. Esta etapa puede abordarse identificando el nivel de zoom específico de cada imagen de entrada o identificando intervalos de niveles de zoom.

En particular, se desarrollaron y analizaron dos modelos de clasificación, el primero fue entrenado en diez clases de nivel de zoom, del 14 al 23, y el segundo modelo de clasificación fue entrenado en dos intervalos de nivel de zoom, el intervalo [14,17] y [18,23]. La tabla III.26 muestra el número de imágenes utilizadas para entrenar y probar estos dos modelos de clasificación. Las imágenes utilizadas fueron seleccionadas de los conjuntos de datos CI-SS_train_stable, CI-SS_test_stable, CI-LS_train_stable y CI-LS_test_stable.

Tabla III.26: Número de imágenes por nivel de zoom utilizadas para el entrenamiento y la evaluación de los clasificadores.

	14	15	16	17	18	19	20	21	22	23
Entrenamiento	252	400	1256	2984	200	591	1080	2268	6406	663
Test	19	52	52	19	44	304	304	304	19	19

La matriz de confusión para la clasificación por nivel de zoom individual se muestra en la tabla III.27. La precisión global de este modelo es del 68.31 %, muy baja.

Tabla III.27: Matriz de confusión para el clasificador por niveles de zoom individuales.

Nivel de zoom	14	15	16	17	18	19	20	21	22	23
14	0	13	5	0	0	0	0	0	1	0
15	0	14	34	2	0	0	0	2	0	0
16	0	0	25	26	0	0	1	0	0	0
17	0	0	1	18	0	0	0	0	0	0
18	0	0	0	33	0	8	2	0	1	0
19	1	0	0	9	0	209	69	12	4	0
20	0	0	0	0	0	12	224	57	11	0
21	0	0	0	2	0	1	6	268	25	2
22	0	0	0	0	0	0	0	2	17	0
23	0	0	0	0	0	0	0	1	18	0

La matriz de confusión para la clasificación por intervalos se muestra en la tabla III.28. Este modelo obtiene una precisión del 96.83 %, que es sustancialmente mayor que la clasificación por nivel de zoom individual. Por lo tanto, se seleccionó este clasificador para incluirlo en la metodología DetDSCI.

4.3.2. Análisis de la metodología DetDSCI

En esta sección, se analiza y compara el rendimiento de la metodología DetDSCI frente a los detectores de referencia CI-LS_Det_stable y CI-SS_Det_stable y un detector de base, Base_Det, entrenado

Tabla III.28: Matriz de confusión para el clasificador por nivel de zoom por intervalos.

Nivel de zoom	[14,17]	[18,23]
[14,17]	134	8
[18,23]	28	966

en todos los datos y niveles de zoom.

Las características de cada modelo son:

- **Base_Det:** es un modelo Faster R-CNN ResNet 101 V1 entrenado en todas las clases de pequeña y gran escala de CI-SS_train_stable y CI-LS_train_stable.
- **CI-LS_Det_stable:** es un modelo Faster R-CNN Inception ResNet V2 entrenado en el conjunto de datos CI-LS_train_stable.
- **CI-SS_Det_stable:** es un modelo Faster R-CNN ResNet 101 V1 con técnicas de DA entrenado en el conjunto de datos CI-SS_train_stable.
- **Metodología DetDSCI:** es la metodología por la cual cada imagen de entrada es clasificada por el clasificador de nivel de zoom y en base a la salida de este clasificador, se selecciona el detector a utilizar entre CI-LS_Det_stable o CI-SS_Det_stable.

Se probaron los cuatro modelos en las imágenes de las clases objetivo, subestación eléctrica de CI-SS_test_stable y aeropuerto de CI-LS_test_stable. Los resultados en términos de TP, FP, FN, Precisión, Recall y F1 se muestran en la tabla III.29.

Tabla III.29: Comparación de rendimiento (%) entre la metodología DetDSCI, Base_Det, CI-LS_Det_stable y CI-SS_Det_stable cuando se testea en la fusión de CI-SS_test_stable y CI-LS_test_stable.

	TP	FP	FN	Precisión	Recall	F1
Base_Det	70	35	44	66.67	61.40	63.93
CI-LS_Det_stable	27	**3**	88	**90.00**	23.48	37.24
CI-SS_Det_stable	71	32	44	68.93	61.74	65.14
Metodología DetDSCI	**83**	24	**32**	77.57	**72.17**	**74.77**

Como se puede ver claramente en la tabla III.29, la metodología DetDSCI supera a Base_Det, CI-SS_Det_stable y CI-LS_Det_stable en todos los aspectos consiguiendo el mayor rendimiento. En concreto, la metodología DetDSCI consigue una mejora en F1 de hasta el 37.53 %. Por lo tanto, se puede concluir que la división entre escalas pequeñas y grandes da mejores resultados. La figura 36 ilustra los resultados de las detecciones obtenidas por Base_Det frente a la metodología DetDSCI.

El tiempo de inferencia del detector a pequeña escala, Faster R-CNN ResNet 101 V1, en una GPU NVIDIA Tesla V100 de 32 GB es de 0.076 segundos, mientras que el detector a gran escala, Faster R-CNN Inception ResNet V2, tarda 0.095 segundos. El clasificador ResNet 50 se ejecuta en 0.0029 segundos. En total, el proceso de la metodología DetDSCI tarda 0.0979 segundos en analizar una imagen de entrada.

Figura 36: Ejemplos de detección obtenidos por el modelo de base, Base_Det (izquierda), y la metodología DetDSCI (derecha).

5. Conclusiones

La detección de infraestructuras críticas en imágenes de satélite es una tarea de gran dificultad debido a las diferencias de escala y a la diversidad de formas, ya que algunas infraestructuras son demasiado pequeñas, por ejemplo, las subestaciones eléctricas, mientras que otras son demasiado grandes, por ejemplo, los aeropuertos. Este trabajo aborda este problema construyendo el conjunto de datos de alta calidad, CI-Dataset, organizado en dos subconjuntos, CI-SS y CI-LS y utilizando la metodología DetDSCI. El proceso de construcción de CI-SS y CI-LS se guio por el rendimiento de los detectores en subestaciones eléctricas y aeropuertos, respectivamente.

La metodología DetDSCI es un enfoque basado en dos etapas que primero identifica el nivel de zoom de la imagen de entrada utilizando un clasificador y luego analiza esa imagen con el modelo de detección correspondiente, CI-LS_Det_stable o CI-SS_Det_stable. La metodología DetDSCI consigue el mayor rendimiento respecto a los detectores de referencia, no sólo en los objetos objetivo sino también en el resto de las clases de infraestructura incluidas en el conjunto de datos.

Como conclusiones, los conjuntos de datos y la metodología propuestos son la mejor solución para abordar el problema de la detección de infraestructuras críticas de escala diferente y distinta en imágenes de teledetección. Este enfoque puede extenderse fácilmente a más infraestructuras críticas.

Los resultados de la investigación realizada en esta etapa del trabajo fueron presentados en un artículo aceptado por la revista *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*.

Capítulo IV

Identificación del cambio en arbustos de alta montaña en imágenes de satélite con *deep learning*

En este capítulo se propone el desarrollo de una metodología basada en una técnica de postprocesamiento de modelos *deep learning* para abordar el problema de la identificación del cambio en arbustos de alta montaña en imágenes de satélite. Con el cambio climático y el aumento de las temperaturas, muchas de estas especies se encuentran en peligro y pueden llegar a extinguirse, por lo que el control del cambio en un periodo de tiempo por parte de personal de conservación de la biodiversidad es de suma importancia.

La identificación del cambio en arbustos de alta montaña supone un reto debido a que la forma que presentan no es semejante entre dos arbustos distintos y también en distintos momentos temporales para el mismo arbusto.

Los modelos de segmentación en imágenes llegan más allá que los modelos de detección de objetos ya que además de localizar donde se encuentra el objeto realizan un estudio de los píxeles que componen al objeto para distinguir la forma de cada uno.

El tipo de arbusto en concreto que se analiza en este capítulo es el enebro, un arbusto de alta montaña presente en Sierra Nevada. Para llevar a cabo su estudio es necesaria la creación de una base de datos especializada y rigurosa, por lo que se ha contado con la ayuda de un experto para la validación del etiquetado mediante un trabajo de campo. Para abordar el problema se ha diseñado la metodología LWJM con la que se hace uso de una técnica de postprocesamiento de modelos *deep learning* que obtiene resultados superiores a los modelos base. Esta metodología está compuesta de dos fases donde en la primera se hace uso de un modelo de segmentación con el que se obtienen los arbustos en cada imagen de entrada. La segunda fase es la encargada de comparar entre dos o más imágenes de la misma zona en diferentes momentos temporales para calcular el cambio que se ha producido a lo largo del tiempo sobre cada arbusto de la imagen.

La metodología LWJM resultante para analizar el cambio en arbustos de alta montaña como el enebro está preparada para ser usada de forma automática por personal de conservación de la biodiversidad en el proyecto LifeWatch ofreciendo una segmentación de cada objeto de calidad.

1. Antecedentes y trabajos relacionados

La segmentación en imágenes con *deep learning* es una de las tareas que más se aborda en grandes competiciones como COCO o ImageNet. Para realizarla, hay tres tipos de segmentación como segmentación semántica, segmentación de instancias y segmentación panóptica. En concreto, esta técnica se va a aplicar a un problema para identificar el cambio producido en distintas temporalidades en arbustos de alta montaña en imágenes de satélite.

1.1. Antecedentes

Varios estudios científicos han demostrado que el cambio climático está alterando la composición, la estructura y el funcionamiento de todos los ecosistemas en cualquier lugar del planeta. La biodiversidad y el clima están ampliamente interconectados porque los cambios en este último afectan directa e indirectamente a la fauna y al ecosistema del medio natural. Una forma de analizar el efecto del cambio climático es estudiar el seguimiento de las especies vegetales especialmente sensibles a los cambios de temperatura en las zonas protegidas, como la alta montaña.

En respuesta al cambio climático, algunas especies vegetales tienden a desplazarse hacia altitudes o latitudes elevadas marcando nuevos patrones de colonización para encontrar condiciones climáticas más favorables [AB98, JMP09, LGM$^+$08, PGD$^+$12]. Debido a la alta sensibilidad climática de este tipo de especies, las montañas pueden considerarse laboratorios naturales ideales para estudiar el cambio global y medir sus efectos.

Según [ZSS$^+$21], la temperatura es la principal causa que restringe el crecimiento de los árboles en las alturas. Por ejemplo, durante las últimas décadas, las especies vegetales de montaña han sufrido cambios significativos en sus distribuciones debido al aumento de la temperatura: en Alaska [LRFS03], Escandinavia [Kul02], los Alpes y otras montañas de Europa occidental [Pau94, LGM$^+$08], y las montañas mediterráneas [PB03, SEDGS03, MJ15]. Estas variaciones en la distribución de las especies pueden caracterizarse en tres comportamientos principales [HP05]: expansión, contracción y desplazamiento del área.

Sin embargo, no todas las especies pueden migrar rápidamente de acuerdo con la velocidad del efecto del cambio climático. Esta dificultad potencial para la migración podría ser especialmente preocupante para los arbustos longevos que tienen una estrategia de persistencia elevada para adaptarse en condiciones extremas de alta montaña (por ejemplo, los enebros de alta montaña) [GZ03, HZ14, MJ15]. Un ejemplo de este tipo de arbusto en alta montaña se puede ver en la figura 37. Adaptando esta estrategia de persistencia, los cambios en la distribución y la estructura poblacional de los arbustos de alta montaña reaccionan más a la variabilidad climática a largo plazo (décadas a siglos) que a la variabilidad climática interanual (años a décadas). Por lo tanto, el comportamiento de los arbustos de alta montaña de larga vida frente al cambio climático convierte a estas especies en centinelas para el seguimiento a largo plazo del cambio global. Por ello, es de suma importancia vigilar los arbustos de alta montaña.

La teledetección por satélite ha proporcionado información espaciotemporal barata, cómoda y continua para observar la superficie terrestre y controlar su cambio a lo largo del tiempo. Una forma de estudiar el impacto del cambio climático en los arbustos de alta montaña es a través de las imágenes RGB de alta resolución recogidas a partir de datos de teledetección por satélite.

El objetivo final de este trabajo es construir un sistema inteligente para el seguimiento de especies de arbustos de alta montaña como el enebro en imágenes de satélite de alta resolución RGB utilizando los modelos de segmentación basados en CNN del estado del arte. Además, se diseñará una metodología para

Figura 37: Imagen de satélite de dos arbustos de enebro en las montañas de Sierra Nevada (Granada).

analizar el cambio que se ha producido entre dos o más instantes de tiempo a través de las imágenes y del uso de la segmentación.

1.2. Trabajos en segmentación de imágenes

La segmentación de imágenes es una de las tareas más desafiantes en visión por computador [Ros76]. Consiste en dividir las imágenes en múltiples segmentos y objetos [Sze10], y desempeña un papel fundamental en una amplia gama de aplicaciones [FP11], incluyendo los vehículos autónomos, el análisis de imágenes médicas, la observación de la tierra, la realidad aumentada, y la videovigilancia, por nombrar algunos.

La segmentación de imágenes puede expresarse como la tarea de clasificar imágenes a nivel de píxel. Existen tres tipos de tareas de segmentación: la segmentación semántica, la segmentación de instancias y la segmentación panóptica. La segmentación semántica consiste en agrupar un conjunto de objetos que pertenecen a la misma categoría (por ejemplo, humanos, árboles, coches, etc.). La segmentación por instancias amplía el alcance de la segmentación semántica para diferenciar entre objetos de la misma categoría (por ejemplo, personas individuales, árboles, etc.). Por su parte, la segmentación panóptica consiste en realizar una partición tanto por categorías como por instancias.

Se han desarrollado varios modelos de segmentación de imágenes en la literatura, desde los primeros modelos (por ejemplo, *k-means clustering* [DMC15], *region-growing* [NN04], etc.) hasta los modelos más avanzados (por ejemplo, *graph cuts* [BVZ01], *active contours* [KWT88], etc.). Recientemente, los modelos de *deep learning* han producido una nueva generación de modelos de segmentación de imágenes con una mejora excepcional del rendimiento, alcanzando a menudo las tasas de precisión más altas en famosos benchmarks de referencia.

1.2.1. Modelos de segmentación semántica

Los modelos de *deep learning* basados en segmentación semántica de imágenes pueden clasificarse en 9 categorías [MBP+20]:

1. Modelos totalmente convolucionales (FCNs). Un FCN consiste únicamente en capas convolucionales sin capas totalmente conectadas para que el modelo produzca un mapa de segmentación espacial en lugar de puntuaciones de clasificación.

2. CNNs con modelos gráficos. Esta categoría consiste en incorporar a las CNNs modelos gráficos probabilísticos (por ejemplo, *Markov Random Fields* (MRFs) y *Conditional Random Fields* (CRFs)) para explotar más contexto en una imagen. [CPK+14] combina CRFs con CNNs. [LSVDHR16] propuso un modelo semántico basado en CRFs contextuales profundos. [LLL+15] desarrolló un modelo que integra información rica en MRFs.

3. Modelos basados en codificador-decodificador. Estos modelos son los más populares de segmentación basados en *deep learning*. Implican un codificador para extraer las características de la imagen y un decodificador para generar un mapa de segmentación. El modelo DeConvNet fue introducido por [NHH15] y consiste en un codificador que contiene capas convolucionales usando VGG-16 y un decodificador que posee capas deconvolucionales. El modelo SegNet fue propuesto por [BKC17] y se diferencia del primer modelo en la forma en que los mapas de características de entrada de menor resolución son remuestreados por el decodificador. Ambos modelos sufren la pérdida de resolución debido al proceso de codificación. Por lo tanto, el modelo HRNet [YCW20] fue creado para controlar esta limitación manteniendo representaciones de alta resolución dentro del proceso de codificación. Algunos modelos utilizan convoluciones transpuestas, como *Stacked Deconvolutional Network* (SDN) [FLW+19], LinkNet [CC17], y el modelo W-Net [XK17], por nombrar algunos. Para segmentar imágenes en el campo médico y biomédico, [RFB15] desarrolló el modelo UNet que consiste en una ruta de contacto utilizada para capturar el contexto de las imágenes, y una ruta de expansión simétrica para una localización precisa. Utilizando este potente modelo se han propuesto varias extensiones.

4. Modelos basados en redes multiescala y piramidales. Uno de los modelos más relevantes propuestos en este contexto es la red piramidal de características (FPN), inicialmente desarrollada para la detección de objetos y luego extendida a tareas de segmentación. Para aprender mejor la representación global del contexto de una imagen, se propuso *Pyramid Scene Parsing Network* (PSPN) [ZSQ+17] utilizando *Residual Network* (ResNet) como extractor de características. Otros modelos han sido creados bajo el mismo marco, incluyendo *Dynamic Multiscale Filters Network* (modelo DM-Net) [HDQ19], *Context Contrasted Network and gated multiscale aggregation* (CCN) [DJS+18], *Adaptive Pyramid Context Network* (APC-Net) [HDZ+19], etc.

5. Modelos convolucionales dilatados. Estos modelos han sido famosos en el campo de la segmentación en tiempo real utilizando registros de vídeo, como los modelos DeepLab [CPK+14, CPK+17], *Dense Upsampling Convolution and Hybrid Dilated Convolution* (DUC-HDC) [WCY+18], *Atrous Spatial Pyramid Pooling* (DenseASPP) [YYZ+18], *Efficient Network* (ENet) [PCKC16], etc.

6. Modelos basados en redes neuronales recurrentes (RNNs). Las RNNs también se pueden utilizar en visión por computador para capturar las relaciones a corto y largo plazo entre los píxeles para mejorar la estimación del mapa de segmentación. Algunos modelos son ReSeg [VCR+16], las redes 2D-LSTM [BBRL15], el modelo Graph-LSTM [LSF+16], las redes neuronales recurrentes asociadas a datos (DA-RNN) [XF17], etc. Sin embargo, estos modelos RNN son más lentos que los modelos CNN debido a la naturaleza secuencial de la computación RNN que no es manejable por la paralelización.

7. Modelos basados en la atención. Los mecanismos de atención han sido ampliamente estudiados en el campo de la visión por computador a lo largo de los años. El mecanismo de atención fue propuesto por primera vez por [CYW+16] para ayudar al modelo a evaluar la importancia de las características en diferentes escalas y posiciones. En el mismo contexto, *Reverse Attention Network* (RAN) [HXW+17] fue desarrollado aplicando un mecanismo de atención inversa para que el modelo aprenda el concepto opuesto. [LXAW18] propuso *Pyramid Attention Network* (PAN) para aprovechar la información contextual global. Se han desarrollado otros modelos basados en la atención, como OCNet [YHG+18], *Expectation-Maximization Attention* (EMANet) [LZW+19], *Discriminative Feature Network* (DFN) [YWP+18], etc.

8. Modelos generativos y entrenamiento adversario. Estos modelos se basan en *Generative Adversarial Networks* (GANs) para realizar la segmentación semántica de imágenes [LCCV16, HTL+18].

9. Modelos CNN con modelos de contorno activo. Estos modelos exploran las interacciones entre las FCN y *Active Contour Models* (MCA) dando lugar a la creación de diferentes modelos, entre los que destacan *Deep Active Contours* [RHBN16], *Deep Active Lesion Segmentation* (DALS) [HHS+20] para imágenes médicas, *Deep Structured Active Contours* (DSAC) [MTK+18], etc.

1.2.2. Modelos de segmentación de instancias

Las CNN basadas en regiones (R-CNN) y sus variantes han demostrado resultados prometedores en aplicaciones de detección de objetos, en particular el modelo Faster R-CNN [RHGS15b]. Como una extensión de Faster R-CNN, se han creado varios modelos para realizar la tarea de segmentación de instancias. El modelo más famoso, que ya ha superado varios retos COCO, es Mask R-CNN [HGDG18]. Este modelo realiza simultáneamente la detección de objetos y la generación de una máscara de segmentación para cada instancia. En el mismo marco, *Path Aggregation Network* (PANet) [LQQ+18] fue propuesto utilizando tanto el modelo Mask-RCNN como FPN. Se han creado otros modelos de segmentación de instancias basados en R-CNN, como DeepMask [PCD15], SharpMask [PLCD16], RetinaMask [FSB19], PolarMask [XSS+20], TensorMask [CGHD19] y CenterMask [LP20].

1.2.3. Modelos de segmentación panóptica

Para controlar la partición por categorías y por instancias de una imagen, se han propuesto varios modelos panópticos, como *Panoptic Feature Pyramid Network* (PFPN) [KGHD19], AdaptIS [SBK19], UPSNet [XLZ+19], OCFusion [LLST20], una red de atención guiada para segmentación panóptica [LCZ+19], *seamless scene segmentation* [PBCK19], *unified panoptic segmentation network* [XLZ+19], *efficient panoptic segmentation* [MV21] y *panoptic Deeplab* [CCZ+20] para la segmentación panóptica en tiempo real.

2. Base de datos y metodología LWJM

En esta sección se explicará el trabajo de campo realizado para la creación de la base de datos LifeWatchJuniperusData (LWJD) y el diseño de la metodología LWJM (LifeWatchJuniperusMethodology) a través de una estrategia de postprocesado de los modelos para el análisis del cambio de arbustos de alta montaña a través de imágenes de satélite con *deep learning*.

2.1. Construcción de la base de datos LWJD

Para construir la base de datos LWJD en las montañas de Sierra Nevada se ha realizado un trabajo de campo en el que se ha desplazado personal experto a las localizaciones para marcar las coordenadas donde existen enebros con un localizador GPS. Posteriormente, esas coordenadas GPS han sido trasladadas al programa QGIS[1] para marcar, por parte de expertos, los puntos que crean la forma de los enebros. De esta manera, esos puntos serán las posteriores coordenadas en las imágenes para marcar el objeto. Así se construye una base de datos adaptada a la tarea de segmentación.

Posteriormente, se ha realizado un trabajo de adquisición de datos a través de la herramienta SAS Planet para la descarga de imágenes de satélite de alta calidad. Estas imágenes están divididas por cuadrículas de 224×224 y además se ha realizado un trabajo manual de filtrado de elementos y de creación de instancias.

La base de datos resultante para llevar a cabo la tarea de *Instance Segmentation* sobre este problema está compuesta como se muestra en la tabla IV.1.

Tabla IV.1: Base de datos LWJD para segmentación de arbustos de alta montaña como los enebros construida en Sierra Nevada.

	Entrenamiento	Validación	Test
Imágenes	567	81	162
Instancias	1138	172	310

2.2. Metodología LWJM

Para la realización de un modelo capaz de segmentar arbustos de alta montaña como los enebros a través de *deep learning* es necesaria la creación de una base de datos. En este problema se usará la construida en las montañas de Sierra Nevada denominada LWJD. En la tabla IV.1 se observa su composición tanto a nivel de imágenes como de instancias.

Gracias a los modelos de segmentación de instancias es posible obtener, para una imagen nueva de entrada, una salida en la que se ofrezcan los objetos encontrados con sus localizaciones y los píxeles que componen cada uno.

La metodología LWJM, que se propone en este trabajo a través de una estrategia de postprocesado de los modelos para el estudio del cambio de arbustos a través de imágenes de satélite con *deep learning*, tiene como fin obtener, de imágenes de distintas fechas en la misma localización, el cambio que se ha producido en un arbusto. De esta forma se podría obtener un modelo, que se ha construido en una zona de Sierra Nevada, para realizar estudios del cambio en diferentes zonas de la misma montaña o en diferentes zonas del globo terráqueo. En la figura 38 se puede observar como actúa la metodología LWJM sobre dos imágenes de fechas diferentes y en la misma localización.

Como entrada a la metodología LWJM se pueden dar múltiples imágenes de una misma localización. Cada una de las imágenes será analizada por el modelo de segmentación y se obtendrá una salida para cada una. Todas las salidas serán analizadas cronológicamente dando para cada arbusto identificado el cambio producido a lo largo de los años. Además, se hará una comparación de superficie para ofrecer una idea de cuánto han crecido o decrecido los arbustos con las ecuaciones:

[1]Programa QGIS: `https://qgis.org/es/site`

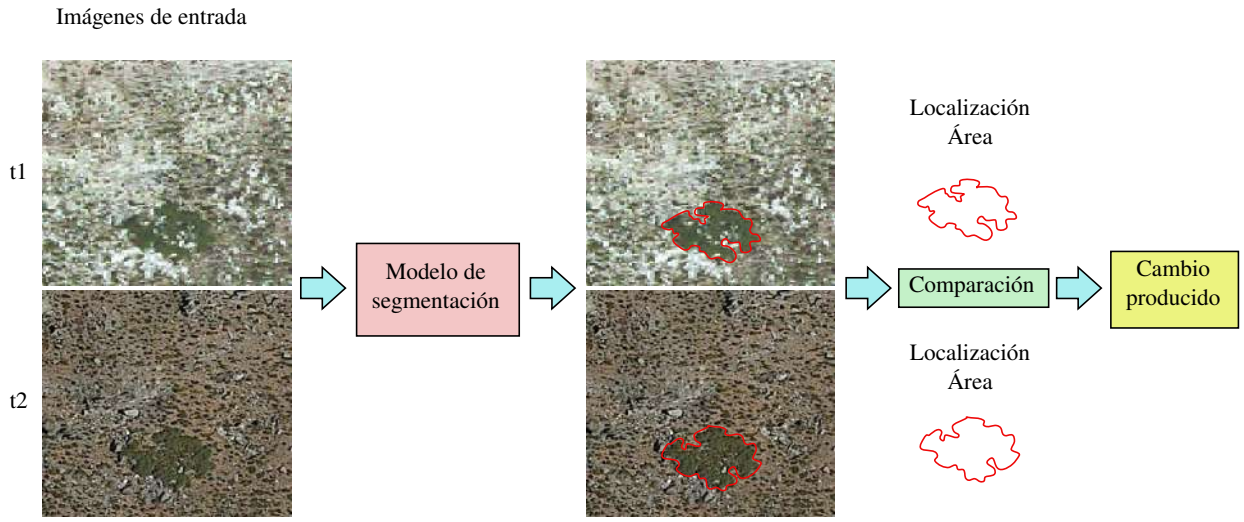


Figura 38: Metodología LWJM para el estudio del cambio en arbustos a través de imágenes de satélite con *deep learning*.

$$Area_{t2} - Area_{t1} = DiferenciaAbsoluta \tag{IV.1}$$

$$(Area_{t2} - Area_{t1})/Area_{t2} = DiferenciaRelativa \tag{IV.2}$$

De esta forma se tendrá conocimiento de lo que ha ocurrido en una zona determinada, tomando medidas si fuese necesario a través de los agentes medioambientales.

3. Análisis experimental

En esta sección se hará un estudio profundizado para el problema de la segmentación de arbustos de alta montaña como los enebros en Sierra Nevada. Como primer punto se analizará la mejor configuración de hiperparámetros para los modelos. Una vez se haya obtenido la configuración óptima de hiperparámetros para este problema se estudiarán diferentes modelos y diferentes arquitecturas. Por último, se realizará un análisis de la metodología LWJM para el estudio del cambio en arbustos.

3.1. Selección de hiperparámetros

Se ha seleccionado Mask R-CNN como modelo base, usando la arquitectura concreta R-50-FPN-1x haciendo uso de la API Detectron2[2]. Para comprobar la configuración que se adapta mejor al problema de segmentación de enebros se van a estudiar los siguientes hiperparámetros:

[2]API Detectron2: `https://github.com/facebookresearch/detectron2`

- *Epochs:* Número de épocas para obtener resultados favorables y que el modelo se estabilice.
- *Decay:* Diferentes técnicas de aplicar o no *decay* como *WarmupCosineLR* o *WarmupMultiStepLR*.
- *Learning rate:* Valores de *learning rate* que obtienen mejores resultados.
- *Batch size:* Valor óptimo de *batch size*.
- *Data augmentation:* Selección del conjunto de técnicas de *data augmentation* que mejoran la robustez del modelo.

Los hiperparámetros iniciales para los primeros modelos han sido de 0.00025 para el *learnin rate*, no se ha usado ninguna técnica de *decay*, el *batch size* ha sido de 128 y no se han aplicado técnicas de *data augmentation*. Los resultados se darán tanto para la detección de *bounding boxes* como para la segmentación.

3.1.1. Análisis del número de épocas

Lo primero será estudiar el valor óptimo para el número de épocas para el entrenamiento del problema. Para ello se seleccionarán diferentes valores y se verá en que punto ocurre el mejor resultado o la estabilización del modelo.

Tabla IV.2: Resultados, en %, de bbox para diferentes valores del número de épocas.

Épocas	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
40	88.13	97.02	96.04	66.78	86.99	75.75	67.38	88.70	81.14
50	91.01	97.97	96.04	67.19	89.14	79.11	67.56	89.43	80.09
60	89.91	97.98	97.00	67.66	89.71	78.88	67.39	**89.85**	80.57
70	93.42	98.01	97.02	67.77	88.48	76.81	67.53	88.84	79.01
80	93.33	98.01	97.02	67.60	87.76	74.31	66.22	86.84	77.62
90	94.77	98.01	97.03	70.24	88.70	78.62	67.72	88.88	79.74
100	95.30	**98.02**	97.02	**70.94**	**89.75**	**79.48**	67.72	88.51	80.32
150	96.23	**98.02**	97.03	68.25	86.70	75.89	68.28	88.55	80.99
200	**97.04**	**98.02**	**98.02**	70.42	88.66	77.30	**69.00**	89.65	**81.74**

Según los resultados obtenidos en la tabla IV.2 y IV.3, como valor inicial para el número de épocas se seleccionará 200 ya que ha obtenido los resultados más favorables en segmentación para los conjuntos de validación y de test. Además, los resultados obtenidos para los bbox también han sido los más resaltados.

3.1.2. Análisis de las técnicas de *decay*

Con el valor de 200 para el número de épocas se estudiará el rendimiento provocado por el uso de las técnicas de *decay*. Para ello se usará el *decay WarmupCosineLR* y el *WarmupMultiStepLR* que hará disminuir el *learing rate* cada 5 épocas con un valor *gamma* del 0.01.

Los resultados obtenidos en las tablas IV.4 y IV.5 muestran como no usar ninguna técnica de *decay* obtiene los mejores resultados en los conjuntos de validación y de test en este problema, por lo que en los siguientes experimentos no se usará esta técnica.

Tabla IV.3: Resultados, en %, de segm para diferentes valores del número de épocas.

Épocas	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
40	75.74	97.00	94.80	62.25	88.07	75.80	62.83	88.70	76.28
50	77.74	97.02	94.93	63.24	**90.30**	76.04	63.98	**90.37**	78.95
60	78.59	97.97	94.91	63.24	89.71	**78.70**	64.03	89.99	**81.29**
70	80.17	97.02	94.98	63.89	88.85	78.62	63.92	89.82	78.97
80	80.89	**98.00**	94.94	63.24	89.26	73.42	62.77	87.85	77.37
90	82.04	97.99	**95.98**	64.78	89.15	77.04	63.71	88.88	77.67
100	82.18	97.02	94.98	65.52	89.98	77.90	64.07	88.55	78.28
150	84.36	97.03	95.01	64.59	87.34	75.67	64.81	88.55	78.86
200	**85.64**	97.03	95.01	**65.85**	89.13	76.91	**66.16**	89.70	80.34

Tabla IV.4: Resultados, en %, de bbox para la aplicación o no de distintas técnicas de *decay*.

Decay	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
No	97.04	**98.02**	**98.02**	**70.42**	88.66	77.30	69.00	**89.65**	**81.74**
WarmupCosineLR	**97.06**	**98.02**	97.03	70.21	88.65	**77.93**	**69.22**	88.77	81.05
WarmupMultiStepLR	68.95	90.21	83.91	64.88	**88.94**	73.87	62.20	87.18	73.67

Tabla IV.5: Resultados, en %, de segm para la aplicación o no de distintas técnicas de *decay*.

Decay	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
No	**85.64**	97.03	95.01	**65.85**	89.13	76.91	**66.16**	**89.70**	80.34
WarmupCosineLR	83.52	**98.00**	**95.02**	65.21	**89.14**	**79.93**	65.21	88.86	**80.41**
WarmupMultiStepLR	64.53	90.31	80.07	62.12	**89.14**	73.00	60.74	87.38	74.27

3.1.3. Análisis del valor de *learning rate*

Una vez estudiado el número de épocas que será de 200, y que no se aplicarán técnicas de *decay*, se van a estudiar distintos valores de *learning rate* como 0.01, 0.0025, 0.001 y 0.00001.

Tabla IV.6: Resultados, en %, de bbox para distintos valores de *learning rate*.

Learning rate	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
0.01	91.01	**98.02**	**98.02**	66.25	83.21	74.39	67.99	86.76	79.80
0.00025	**97.04**	**98.02**	**98.02**	**70.42**	88.66	**77.30**	**69.00**	**89.65**	**81.74**
0.0001	96.40	98.01	97.03	68.30	88.46	74.11	67.65	88.57	79.29
0.00001	77.42	95.25	91.82	64.15	**88.72**	75.83	62.82	86.23	74.84

Las tablas IV.6 y IV.7 muestran como el valor de *learning rate* que obtiene los resultados más estables es 0.0025, y que será el que se use en los siguientes experimentos.

Tabla IV.7: Resultados, en %, de segm para distintos valores de *learning rate*.

Learning rate	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
0.01	**88.11**	96.97	95.94	64.55	84.35	74.83	66.15	86.76	**80.42**
0.00025	85.64	**97.03**	95.01	**65.85**	89.13	**76.91**	**66.16**	**89.70**	80.34
0.0001	82.54	**97.03**	**95.98**	64.19	**89.78**	76.42	64.19	88.81	79.49
0.00001	69.51	94.34	87.36	62.24	88.79	73.24	61.29	87.64	75.22

3.1.4. Análisis del valor de *batch size*

Con el valor del *learning rate* ajustado a 0.0025, el número de épocas a 200 y sin aplicar las técnicas de *decay*, se va a estudiar la influencia del parámetro *batch size*. Se van a realizar experimentos con los valores 32, 68, 128 y 512.

Tabla IV.8: Resultados, en %, de bbox para distintos valores de *batch size*.

Batch size	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
32	96.38	**98.02**	97.03	69.82	87.28	**78.88**	68.35	88.81	80.37
64	96.68	**98.02**	97.03	69.39	85.85	78.50	69.56	88.62	81.76
128	**97.04**	**98.02**	**98.02**	**70.42**	**88.66**	77.30	69.00	**89.65**	81.74
512	96.23	**98.02**	97.03	69.81	86.83	77.31	**69.78**	88.79	**83.22**

Tabla IV.9: Resultados, en %, de segm para distintos valores de *batch size*.

Batch size	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
32	86.30	97.03	**96.01**	**65.85**	88.91	75.79	65.33	88.87	77.55
64	86.22	97.03	95.99	64.43	87.47	75.93	65.85	89.56	79.29
128	85.64	97.03	95.01	**65.85**	**89.13**	76.91	66.16	89.70	80.34
512	**86.95**	**98.01**	95.98	65.30	87.19	**77.59**	**66.25**	**89.74**	**80.44**

Según los resultados obtenidos en las tablas IV.8 y IV.9 el valor 128 será el usado por defecto ya que obtiene resultados superiores en el conjunto de validación.

3.1.5. Análisis del impacto de las técnicas de *data augmentation*

En este punto, los valores ajustados han sido el número de épocas a 200, no usar técnicas de *decay*, el valor de *learning rate* a 0.00025 y el valor de *batch size* a 128. Los siguientes experimentos van a comprobar si el uso de técnicas de *data augmentation* ofrecen mejoras sobre los modelos.

Para ello, se va a estudiar la influencia de las técnicas *RandomFlip*, *RandomRotation*, *Resize*, *ResizeScale*, *FixedSizeCrop*, *RandomCrop*, *RandomLighting*, *RandomBrightness*, *RandomContrast* y *RandomSaturation*.

Tabla IV.10: Resultados, en %, de bbox para el uso de técnicas de *data augmentation*.

Data	Entrenamiento			Validación			Test		
Augmentation	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
No	**97.04**	**98.02**	**98.02**	**70.42**	**88.66**	77.30	69.00	89.65	81.74
Sí	96.45	**98.02**	**98.02**	70.37	87.74	**78.45**	**69.25**	**89.84**	**82.13**

Tabla IV.11: Resultados, en %, de segm para el uso de técnicas de *data augmentation*.

Data	Entrenamiento			Validación			Test		
Augmentation	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
No	85.64	**97.03**	95.01	65.85	**89.13**	76.91	**66.16**	89.70	**80.34**
Sí	**87.44**	**97.03**	**95.99**	**65.94**	88.95	**78.36**	65.79	**89.99**	80.23

En las tablas IV.10 y IV.11 se muestra como el uso de técnicas de *data augmentation* ofrece mejores resultados a los modelos, por lo que en los siguientes análisis se usarán estas técnicas.

3.2. Estudio de diferentes modelos y arquitecturas

El estudio sobre los hiperparámetros realizado para el problema de la segmentación de arbustos de alta montaña como el enebro ha dado como resultado que el modelo será entrenado por defecto 200 épocas, sin aplicar técnicas de *decay*, usando como valor de *learning rate* 0.00025, un *batch size* de 128 y aplicando técnicas de *data augmentation*.

Son tres los diferentes modelos del estado del arte que se van a analizar en este problema, Mask R-CNN, TensorMask y CenterMask con diferentes arquitecturas para cada uno de ellos.

3.2.1. Análisis de diferentes arquitecturas para el modelo Mask R-CNN

Se va a realizar un análisis para diferentes arquitecturas del modelo Mask R-CNN.

En la tabla IV.12 y IV.13 se muestran los resultados obtenidos para diferentes arquitecturas de Mask R-CNN. La arquitectura con el mejor rendimiento ha sido R_50_FPN_1x y X_101_32x8d_FPN_3x al obtener resultados muy similares. En este caso se seleccionará el modelo X_101_32x8d_FPN_3x como mejor al obtener el mayor resultado en el conjunto de test.

3.2.2. Análisis de diferentes arquitecturas para el modelo TensorMask

Se va a realizar un análisis para diferentes arquitecturas del modelo TensorMask.

En la tabla IV.14 y IV.15 se muestran los resultados obtenidos para diferentes arquitecturas de TensorMask. La arquitectura con el mejor rendimiento ha sido R_50_FPN_6x al obtener los mejores resultados para validación y test.

Tabla IV.12: Resultados, en %, de bbox para distintas arquitecturas de Mask R-CNN.

Modelos	Entrenamiento			Validación			Test		
Mask R-CNN	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
R_101_C4_3x	96.76	98.01	97.02	70.08	89.21	78.64	70.20	88.34	79.37
R_101_DC5_3x	97.08	98.01	97.03	71.22	88.27	80.10	70.80	89.06	81.39
R_101_FPN_3x	88.66	97.03	96.03	67.45	**89.54**	**81.92**	71.00	87.87	80.85
R_50_C4_1x	96.00	97.03	97.03	70.01	85.55	77.11	69.36	88.89	82.16
R_50_C4_3x	96.27	97.03	97.03	70.08	87.32	78.18	69.48	87.30	80.69
R_50_DC5_1x	96.37	98.00	97.02	69.51	87.47	75.84	70.43	88.95	80.60
R_50_DC5_3x	96.90	98.01	97.03	**71.41**	86.95	79.26	70.45	89.76	80.52
R_50_FPN_1x	96.45	**98.02**	**98.02**	70.37	87.74	78.45	69.25	**89.84**	82.13
R_50_FPN_3x	97.21	**98.02**	**98.02**	71.29	87.76	81.23	70.11	89.48	81.31
X_101_32x8d_FPN_3x	**97.61**	**98.02**	**98.02**	69.99	86.77	78.06	**71.36**	89.26	**82.82**

Tabla IV.13: Resultados, en %, de segm para distintas arquitecturas de Mask R-CNN.

Modelos	Entrenamiento			Validación			Test		
Mask R-CNN	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
R_101_C4_3x	86.99	98.00	95.98	65.42	**90.43**	77.14	65.34	89.28	78.95
R_101_DC5_3x	91.02	98.00	95.99	66.23	90.03	75.57	67.40	89.10	**82.85**
R_101_FPN_3x	70.85	89.26	79.68	67.45	89.54	**81.92**	67.35	88.78	80.28
R_50_C4_1x	84.11	97.02	94.84	64.74	88.21	78.78	64.95	88.90	78.69
R_50_C4_3x	85.97	96.97	95.95	66.43	89.10	80.64	65.72	88.41	79.24
R_50_DC5_1x	90.18	97.03	95.03	66.55	88.94	76.91	67.30	89.16	81.58
R_50_DC5_3x	89.87	97.03	96.01	**67.79**	88.36	80.56	67.19	89.85	81.61
R_50_FPN_1x	87.44	97.03	95.99	65.94	88.95	78.36	65.79	**89.99**	80.23
R_50_FPN_3x	86.61	97.03	96.01	66.71	89.06	77.34	67.14	89.52	80.72
X_101_32x8d_FPN_3x	**91.66**	**98.02**	**96.02**	66.96	88.74	77.51	**68.60**	89.26	81.60

Tabla IV.14: Resultados, en %, de bbox para distintas arquitecturas de TensorMask.

Modelos	Entrenamiento			Validación			Test		
TensorMask	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
R_50_FPN_1x	**96.91**	**99.00**	**98.01**	65.52	88.23	75.52	65.21	88.48	75.64
R_50_FPN_6x	91.49	98.00	96.04	**68.54**	**89.94**	**77.55**	**69.91**	**90.79**	**80.87**

Tabla IV.15: Resultados, en %, de segm para distintas arquitecturas de TensorMask.

Modelos	Entrenamiento			Validación			Test		
TensorMask	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
R_50_FPN_1x	**88.73**	**97.91**	**95.83**	57.98	**90.45**	65.31	58.61	87.34	68.83
R_50_FPN_6x	79.53	97.89	93.55	**60.87**	90.23	**66.97**	**63.06**	**90.86**	**77.53**

3.2.3. Análisis de diferentes arquitecturas para el modelo CenterMask

Se va a realizar un análisis para diferentes arquitecturas del modelo CenterMask.

Tabla IV.16: Resultados, en %, de bbox para distintas arquitecturas de CenterMask.

Modelos	Entrenamiento			Validación			Test		
CenterMask	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
lite_Mv2_FPN_ms_4x	93.30	97.03	96.04	63.44	87.79	72.52	63.29	89.74	73.49
lite_V_19_slim_dw_eSE_FPN_ms_4x	88.51	96.88	95.03	60.13	88.60	69.80	60.15	87.87	73.96
lite_V_19_slim_eSE_FPN_ms_4x	88.15	97.02	96.03	57.63	86.17	70.67	60.66	87.64	71.57
lite_V_39_eSE_FPN_ms_4x	94.38	97.03	96.04	63.43	88.10	74.08	66.97	90.63	76.00
R_50_FPN_ms_3x	95.69	**98.02**	**97.03**	66.64	86.97	73.74	66.67	89.27	74.98
R_101_FPN_ms_3x	**96.31**	98.01	**97.03**	71.20	88.78	80.26	68.06	88.77	77.89
V_39_eSE_dcn_FPN_ms_3x	91.60	98.01	**97.03**	65.75	89.53	72.19	67.93	90.73	80.51
V_39_eSE_FPN_ms_3x	91.73	98.01	97.02	68.66	89.49	77.80	68.32	90.19	81.03
V_57_eSE_dcn_FPN_ms_3x	93.68	97.03	**97.03**	66.36	88.99	75.19	66.66	90.57	76.93
V_57_eSE_FPN_ms_3x	93.64	97.03	**97.03**	**71.38**	**92.32**	**81.56**	**71.23**	**93.24**	83.25
V_99_eSE_dcn_FPN_ms_3x	93.32	98.01	97.02	67.40	87.52	77.12	67.42	89.23	78.65
V_99_eSE_FPN_ms_3x	93.76	97.03	97.02	67.97	87.56	77.38	70.46	91.52	**83.52**

Tabla IV.17: Resultados, en %, de segm para distintas arquitecturas de CenterMask.

Modelos	Entrenamiento			Validación			Test		
CenterMask	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
lite_Mv2_FPN_ms_4x	82.04	96.99	94.88	59.09	87.74	67.43	60.96	89.64	70.93
lite_V_19_slim_dw_eSE_FPN_ms_4x	77.20	96.93	94.22	56.29	88.35	60.98	58.07	87.59	70.10
lite_V_19_slim_eSE_FPN_ms_4x	77.76	97.01	94.65	56.15	87.84	62.26	58.71	87.59	72.66
lite_V_39_eSE_FPN_ms_4x	86.19	97.03	96.00	61.26	89.15	70.97	63.51	90.66	75.06
R_50_FPN_ms_3x	89.15	97.03	96.02	64.03	88.90	74.57	65.09	88.92	77.96
R_101_FPN_ms_3x	**92.22**	97.03	**97.02**	67.56	90.34	79.05	65.72	88.74	78.92
V_39_eSE_dcn_FPN_ms_3x	84.77	97.03	95.98	63.86	89.53	74.25	65.24	91.38	80.55
V_39_eSE_FPN_ms_3x	86.13	**97.98**	97.00	64.76	91.78	75.24	65.34	90.89	79.46
V_57_eSE_dcn_FPN_ms_3x	85.76	97.03	96.77	64.14	88.99	74.02	64.98	90.62	78.64
V_57_eSE_FPN_ms_3x	86.91	97.02	95.99	**68.93**	**93.14**	**80.89**	67.49	**93.24**	**81.44**
V_99_eSE_dcn_FPN_ms_3x	84.99	97.78	95.97	64.71	89.46	75.28	64.85	89.33	78.17
V_99_eSE_FPN_ms_3x	86.80	97.02	95.97	64.37	88.20	73.31	**67.62**	91.51	80.72

En la tabla IV.16 y IV.17 se muestran los resultados obtenidos para diferentes arquitecturas de CenterMask. La arquitectura con el mejor rendimiento ha sido V_57_eSE_FPN_ms_3x al obtener los mejores resultados tanto para validación como para test.

3.2.4. Comparativa de los mejores modelos

Una vez analizadas las diferentes arquitecturas para cada uno de los tres modelos seleccionados del estado del arte, se comparará entre cada una de las mejores arquitecturas para seleccionar un modelo final con los mejores resultados.

En la tabla IV.18 y IV.19 se muestran los resultados de las mejores arquitecturas para los modelos. Por lo tanto, el modelo que mejor resultados ha obtenido en el problema de la segmentación de arbustos

Tabla IV.18: Resultados, en %, de bbox para las mejores arquitecturas de cada uno de los modelos.

Modelo	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
Mask R-CNN X_101_32x8d_FPN_3x	**97.61**	**98.02**	**98.02**	69.99	86.77	78.06	**71.36**	89.26	82.82
TensorMask R_50_FPN_6x	91.49	98.00	96.04	68.54	89.94	77.55	69.91	90.79	80.87
CenterMask V_57_eSE_FPN_ms_3x	93.64	97.03	97.03	**71.38**	**92.32**	**81.56**	71.23	**93.24**	**83.25**

Tabla IV.19: Resultados, en %, de segm para las mejores arquitecturas de cada uno de los modelos.

Modelo	Entrenamiento			Validación			Test		
	AP	AP50	AP75	AP	AP50	AP75	AP	AP50	AP75
Mask R-CNN X_101_32x8d_FPN_3x	**91.66**	**98.02**	**96.02**	66.96	88.74	77.51	**68.60**	89.26	**81.60**
TensorMask R_50_FPN_6x	79.53	97.89	93.55	60.87	90.23	66.97	63.06	90.86	77.53
CenterMask V_57_eSE_FPN_ms_3x	86.91	97.02	95.99	**68.93**	**93.14**	**80.89**	67.49	**93.24**	81.44

como el enebro ha sido el modelo CenterMask con la arquitectura V_57_eSE_FPN_ms_3x para el conjunto de validación y el modelo Mask R-CNN con la arquitectura X_101_32x8d_FPN_3x para el conjunto de test. En la siguiente sección para el estudio de la metodología LWJM se compararán los resultados para ambos modelos al tener resultados similares.

3.3. Estudio de la metodología LWJM para observar el cambio en arbustos de alta montaña

En esta sección se estudiarán los resultados de la metodología LWJM para observar el cambio en arbustos de alta montaña. Para ello se usarán los dos modelos que mejor resultado han obtenido en el estudio anterior, Mask R-CNN X_101_32x8d_FPN_3x y CenterMask V_57_eSE_FPN_ms_3x. Se han descargado imágenes de 225×225 píxeles de la misma localización en Sierra Nevada, las cuales no se han usado para entrenamiento, en dos años diferentes, 2010 y 2020.

Para cada una de las imágenes se mostrará, empezando de izquierda a derecha, la imagen original, la imagen con la segmentación realizada con el modelo CenterMask y la imagen con la segmentación realizada con el modelo Mask R-CNN.

Los resultados para la figura 39 son:

- CenterMask obtiene tres enebros en ambos años donde para el caso del elemento central las dimensiones han cambiado aumentando el tamaño.
- El modelo Mask R-CNN solo obtiene todos los arbustos para el año 2020 ya que la calidad de la imagen de 2010 es inferior.
- Las áreas de cada arbusto empezando de izquierda a derecha, para CenterMask han sido en 2010 de 744, 837 y 456 y en 2020 de 983, 1272 y 358.
- Las áreas de cada arbusto empezando de izquierda a derecha, para Mask R-CNN han sido en 2010 de 747 y en 2020 de 1000, 990, 434.
- En el caso de CenterMask si se obtiene la DiferenciaAbsoluta para cada arbusto, comenzando de izquierda a derecha, se obtienen unos valores de 239, 435 y -98, y una DiferenciaRelativa de 24.31 %,

Figura 39: Original (izquierda), CenterMask (centro) y Mask R-CNN (derecha) para analizar el cambio producido entre 2010 (superior) y 2020 (inferior) en arbustos en Sierra Nevada.

34.20 % y -27.37 %. Se puede decir en base a estos resultados que dos de los arbustos han crecido en este periodo de diez años, mientras que uno se ha reducido.

- En el caso de Mask R-CNN si se obtiene la DiferenciaAbsoluta para el arbusto identificado, se obtienen un valor de 253, y una DiferenciaRelativa de 25.30 %. Se puede decir en base a estos resultados que el arbusto ha crecido en este periodo de diez años.

Los resultados para la figura 40 son:

- CenterMask obtiene cuatro enebros en ambos años en la parte inferior y para el 2010 ha obtenido uno en la parte superior. Se puede pensar que ese arbusto ha disminuido su tamaño con el paso de los años.
- El modelo Mask R-CNN solo obtiene arbustos para el año 2020 ya que la calidad de la imagen de 2010 es inferior.
- Las áreas de cada arbusto empezando de izquierda a derecha, para CenterMask han sido en 2010 de 687, 1739, 579, 1019 y 867 y en 2020 de 782, 554, 1199 y 1068.
- Las áreas de cada arbusto empezando de izquierda a derecha, para Mask R-CNN han sido en 2020 de 903, 715, 1141 y 977.
- En el caso de CenterMask si se obtiene la DiferenciaAbsoluta para cada arbusto, comenzando de izquierda a derecha, se obtienen unos valores de 95, 25, 180 y 201, y una DiferenciaRelativa de

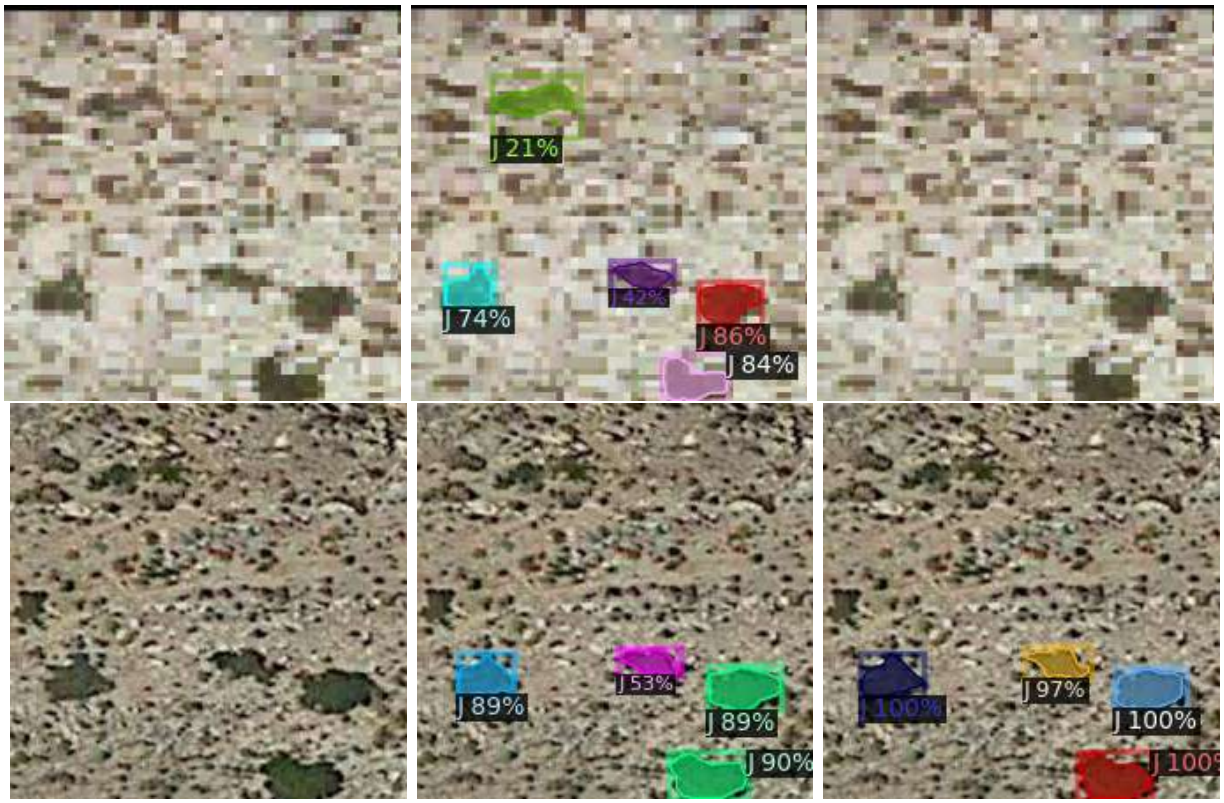


Figura 40: Original (izquierda), CenterMask (centro) y Mask R-CNN (derecha) para analizar el cambio producido entre 2010 (superior) y 2020 (inferior) en arbustos en Sierra Nevada.

12.15 %, 4.51 %, 15.01 % y 18.82 %. Se puede decir en base a estos resultados que los arbustos han crecido en este periodo de diez años.

- En el caso de Mask R-CNN la DiferenciaAbsoluta y DiferenciaRelativa no se pueden calcular al no tener Mask R-CNN resultados para el año 2010.

Los resultados para la figura 41 son:

- CenterMask ha obtenido dos nuevos arbustos para el 2020 que no se aprecian en el 2010 y además el tamaño de estos ha aumentado.
- Mask R-CNN presenta un nuevo arbusto en comparación, y un aumento considerable del tamaño.
- Las áreas de cada arbusto empezando de izquierda a derecha, para CenterMask han sido en 2010 de 773, 957, 433, 696 y 967 y en 2020 de 1080, 192, 571, 1380, 1225, 706 y 684.
- Las áreas de cada arbusto empezando de izquierda a derecha, para Mask R-CNN han sido en 2010 de 731, 1274, 814, 471 y 523 y en 2020 de 1238, 599, 1468, 1101, 664 y 669.
- En el caso de CenterMask si se obtiene la DiferenciaAbsoluta para cada arbusto (salvo el arbusto que ha desaparecido entre una fecha y otra), comenzando de izquierda a derecha y de arriba a abajo, se obtienen unos valores de 307, 423, 792, 10 y -283, y una DiferenciaRelativa de 28.43 %, 30.65 %, 64.65 %, 1.42 % y -41.37 %.

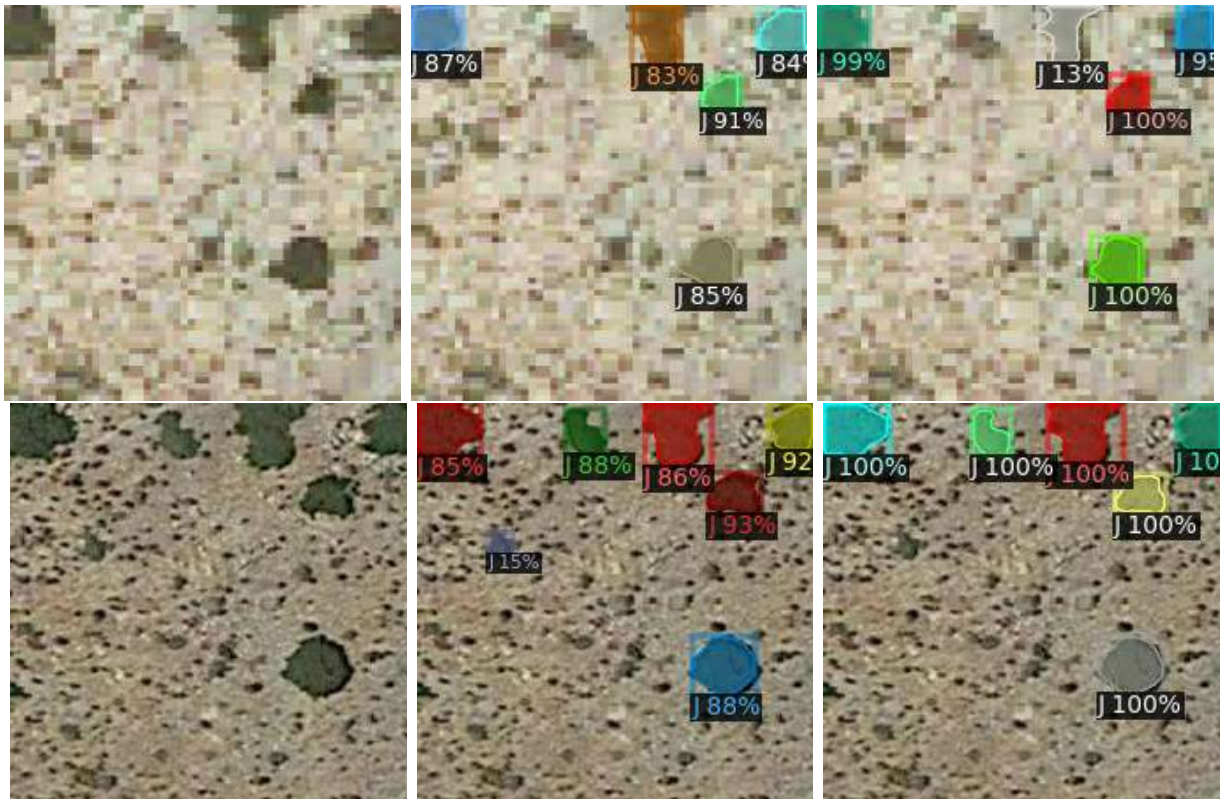


Figura 41: Original (izquierda), CenterMask (centro) y Mask R-CNN (derecha) para analizar el cambio producido entre 2010 (superior) y 2020 (inferior) en arbustos en Sierra Nevada.

- En el caso de Mask R-CNN si se obtiene la DiferenciaAbsoluta para el arbusto identificado, se obtiene un valor de 507, 194, 287, 193 y 146, y una DiferenciaRelativa de 40.95 %, 13.22 %, 26.07 %, 29.07 % y 21.82 %.

4. Conclusiones

Este trabajo ha presentado la metodología LWJM basada en *deep learning* para el análisis del cambio en arbustos de alta montaña como es el enebro. Esta metodología está compuesta de una técnica de postprocesado de los modelos *deep learning* en la que con la salida ofrecida por los modelos se realiza una comparación del área del objeto en diferentes fechas. Se consideró como caso de estudio el problema de la conservación de ciertas especies de arbustos ya que con el cambio climático estas están expuestas a desplazarse para conseguir condiciones climáticas más favorables.

Se diseñó una base de datos llamada LWJD compuesta de enebros en una zona de Sierra Nevada, construida a partir de trabajo de campo en el que un experto validó la localización del arbusto de forma presencial con un localizador GPS.

Los experimentos realizados estudiaron diferentes arquitecturas para los modelos Mask R-CNN, CenterMask y TensorMask. Además, se seleccionaron los hiperparámetros adecuados para el problema de la segmentación de arbustos a través de *Instance Segmentation*.

Con la metodología LWJM se pudo comprobar el cambio sufrido en un determinado arbusto a través

de imágenes de diferentes años controlando la conservación de la especie que se puede ver en peligro por el cambio climático.

Capítulo V

Conclusiones y futuras líneas de investigación

Este capítulo final tiene por objeto ofrecer una visión general del trabajo desarrollado a lo largo de la presente tesis, destacando los logros más importantes, enumerando las distintas publicaciones realizadas y resumiendo las líneas de investigación abiertas en las que se continuará trabajando en el futuro.

La primera fase del trabajo consistió en una revisión amplia de la bibliografía especializada en relación con el tema de estudio: el diseño de estrategias de pre y postprocesamiento para modelos *deep learning*. Esto permitió conocer el estado del arte para las diferentes tareas de visión por computador como clasificación, detección y segmentación y los diferentes tipos de datos que se pueden usar. Con el objetivo del diseño de estrategias novedosas de pre y postprocesado para modelos *deep learning* en los ámbitos de seguridad y biodiversidad, se han definido los siguientes objetivos específicos:

- Estrategia de postprocesamiento para la detección de armas en el ámbito de la seguridad para distintos objetos pequeños que se manejan de forma similar a un arma.
- Estrategia de preprocesamiento para la detección de infraestructuras críticas en imágenes de satélite en el ámbito de la seguridad.
- Estrategia de postprocesamiento para la detección del cambio en arbustos de alta montaña en imágenes de satélite en el ámbito de la biodiversidad.

En la siguiente sección se ofrece una visión general de los resultados obtenidos, asociados a cada uno de esos objetivos.

1. Objetivos y resultados alcanzados

Los objetivos que se plantearon al inicio de la presente tesis han generado como resultado tres estrategias de pre y postprocesado para modelos *deep learning* con el objeto de resolver tres problemas diferentes en los ámbitos de la seguridad y la biodiversidad. Atendiendo a los tres problemas, los resultados alcanzados son los resumidos a continuación.

1.1. Detección de armas

En el segundo capítulo se diseñó la primera estrategia de postprocesado para el problema de la detección de armas. En este problema se realizó la detección de objetos pequeños que son manejados similarmente a un arma. En el primer paso se estudiaron los trabajos relacionados con la detección de objetos y las técnicas de binarización como OVA y OVO para aplicarlas en imágenes. Debido a que en las colas de los bancos y en las joyerías se usan objetos como el smartphone, billete, monedero y tarjeta, estos objetos han formado parte de la base de datos que se creó junto con las pistolas y los cuchillos, la base de datos Sohas_weapon. Además, se diseñó una metodología llamada ODeBiC basada en *deep learning* con la que se abordó el problema. Esta metodología se compone de dos etapas, una primera en la que se analiza un frame con un detector entrenado con las seis clases de objetos de forma que las regiones que se proponen son enviadas a la siguiente etapa. En esta segunda etapa, cada una de las regiones es analizada por cada uno de los clasificadores binarios, de forma que a la salida de todos ellos se aplica un método de agregación para obtener la clase final.

Los resultados de la experimentación realizada en el problema de la detección de armas fueron los siguientes:

- Incluir en la base de datos, Sohas_weapon, objetos pequeños que se manejan de forma similar a un arma reducía el número de falsos positivos que comete un modelo. De esta forma, el modelo distingue entre los objetos más comunes que se usan en determinadas situaciones mitigando los falsos positivos y las llamadas erróneas a los cuerpos de seguridad.
- Usar un detector entrenado con todas las clases para hacer una búsqueda de las regiones proporcionaba un mejor resultado ya que las regiones propuestas buscan coincidir con un objeto ya conocido, de forma que esta es una búsqueda guiada y especializada.
- El uso de las técnicas de binarización como estrategia de postprocesamiento proporcionaba una confianza mayor al tratarse de clasificadores con conocimiento experto para así discernir entre objetos muy similares o que se manejan de una forma semejante. Esto provocaba una reducción considerable de falsos positivos ya que de todas las salidas proporcionadas por los distintos clasificadores binarios se realizaba un método de agregación para obtener la clase final.
- El uso de un modelo de detección como Faster R-CNN para obtener las regiones propuestas y su posterior análisis con clasificadores binarios basados en ResNet 101 daban lugar a un sistema que puede trabajar en tiempo real en una GPU de categoría media. Por lo tanto, el sistema resultante puede ser implementado con un bajo costo a través de cámaras de videovigilancia.

El sistema resultante se puede usar en un establecimiento para prevenir posibles atracos, conociendo sus limitaciones al necesitar ser ampliado para poder prescindir de otro tipo de seguridad y ser completamente autónomo.

1.2. Detección de infraestructuras críticas

En el tercer capítulo se realizó una estrategia de preprocesado para el problema de la detección de infraestructuras críticas en imágenes de satélite. El primer paso fue estudiar los trabajos relacionados para la detección de objetos en imágenes de satélite. Seguidamente se diseñó la metodología DetDSCI dividida en dos etapas, donde en la primera, a través de un clasificador de niveles de zoom, se calcula el nivel de zoom

de una imagen de entrada. Con la salida de esa primera etapa se selecciona un detector especializado en función del nivel de zoom para que se realice la detección de determinadas clases. Para resolver el problema se creó un dataset especializado en infraestructuras críticas denominado CI-dataset, el cual se construyó en diferentes pasos guiados por el rendimiento del modelo de detección.

Los resultados de la experimentación realizada en el problema de la detección de infraestructuras críticas fueron los siguientes:

- El conjunto de datos CI-dataset construido se dividió en función de los niveles de zoom, dando lugar a dos subconjuntos, uno para las infraestructuras de escala grande y otro para las infraestructuras de escala pequeña. De esta forma las infraestructuras objetivo se dividieron para compartir características y que los modelos fuesen especializados.
- La creación de dos detectores especializados en diferentes escalas para las infraestructuras objetivo obtuvo resultados superiores a un detector base en el que se entrenaron todas las clases de objetos sin importar su nivel de zoom.
- Un clasificador *deep learning* fue capaz de distinguir, con un alto nivel de precisión, los diferentes niveles de zoom de las imágenes de satélite, especialmente si este clasificador está compuesto por dos intervalos de niveles en vez de niveles individuales.
- El desarrollo de una técnica de preprocesamiento en la metodología DetDSCI hizo posible la detección de infraestructuras críticas en imágenes de satélite sin tener que seleccionar el nivel de zoom manualmente.

El sistema para detectar infraestructuras puede ser usado sin necesidad de conocimientos previos y conseguir distinguir entre diferentes infraestructuras.

1.3. Detección del cambio en arbustos de alta montaña

En el cuarto capítulo se realizó una estrategia de postprocesado para el problema de la identificación del cambio en arbustos de alta montaña, como es el caso del enebro en las montañas de Sierra Nevada, en imágenes de satélite de diferentes temporalidades, causado por el cambio climático y el aumento de las temperaturas. Para llevar a cabo esta tarea se hizo uso de los modelos del estado del arte para segmentación de objetos. Existe tres tipos de segmentación en imágenes y en este problema se seleccionó segmentación de instancias para llevarlo a cabo. Como primer paso se construyó una base de datos denominada LWJD, la cual fue validada por expertos a través de un trabajo de campo con un localizador GPS. Seguidamente se diseñó la metodología LWJM, la cual hace uso de dos o más imágenes de diferentes temporalidades en el mismo punto. Estas imágenes fueron analizadas por el modelo de segmentación obteniendo los enebros en la imagen y el área que ocupan. El siguiente paso fue la comparación de las salidas del modelo para comprobar si durante el paso del tiempo los arbustos han sufrido cambios y si pueden verse en peligro. Una tarea realizada en este capítulo fue el análisis experimental para ajustar los hiperparámetros del modelo de segmentación al problema en concreto y el estudio de diferentes modelos y arquitecturas.

Los resultados de la experimentación realizada en el problema de la detección del cambio de los arbustos fueron los siguientes:

- La construcción del dataset LWJD, llevado a cabo con la colaboración del trabajo de campo realizado por un experto, proporciona un conocimiento a la comunidad científica de gran valor para realizar

estudios posteriores. Este dataset permite la creación de modelos de segmentación y detección al estar etiquetado manualmente para ambas tareas.

- Los modelos de segmentación de instancias permitieron obtener un número de objetos en una imagen pudiendo discernir entre varios de ellos, al contrario de otro tipo de segmentaciones como la segmentación semántica en la que todos los objetos se seleccionan como uno solo.
- A partir de un modelo de segmentación es posible obtener la localización y el área que ocupa un objeto en una imagen. De esta forma se pudo obtener el tamaño que ocupa un arbusto de alta montaña de forma sencilla en imágenes de satélite.
- El sistema diseñado para este problema, la metodología LWJM con una técnica de postprocesamiento, permitió obtener la información del cambio de distintos arbustos de alta montaña en diferentes fechas de una forma sencilla aportando un gran conocimiento para dar datos de utilidad a los agentes encargados de conservar la biodiversidad.

El sistema que analiza el cambio en arbustos de alta montaña está preparado para ser usado de forma automática por personal de conservación de la biodiversidad en el proyecto LifeWatch.

2. Publicaciones

Producto de los resultados que se han alcanzado, enumerados en la sección previa, esta tesis tiene asociadas las publicaciones enumeradas a continuación:

- Los resultados del segundo capítulo fueron condensados en el siguiente articulo de investigación en una revista del primer cuartil en la categoría *Computer Science, Artificial Intelligence* con ranking 16/139 y factor de impacto (JCR 2020) 8.038:

  "Pérez-Hernández, F., Tabik, S., Lamas, A., Olmos, R., Fujita, H., & Herrera, F. (2020). *Object detection binary classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance. Knowledge-Based Systems*, 194, 105590."

- Los resultados del tercer capítulo fueron condensados en el siguiente articulo de investigación en una revista del segundo cuartil en las categorías *Imaging Science and Photographic Technology* y *Remote Sensing* con ranking 10/29 y 15/32, respectivamente, y con factor de impacto (JCR 2020) 3.784:

  "Pérez-Hernández, F., Rodríguez-Ortega, J., Benhammou, Y., Herrera, F., & Tabik, S. (2021). *CI-dataset and DetDSCI methodology for detecting too small and too large critical infrastructures in satellite images: Airports and electrical substations as case study. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14, 12149-12162."

- La publicación asociada a los resultados del cuarto capítulo se encuentra en progreso actualmente. Hasta el momento se ha diseñado la base de datos, se ha creado una metodología específica para el problema y se ha realizado la experimentación por lo que se encuentra en proceso de escritura.

3. Líneas de trabajo futuras

A partir del trabajo desarrollado durante la elaboración de esta tesis en problemas diferentes se han ido abriendo nuevas vías de investigación que pueden dar lugar a líneas de trabajo futuras. A continuación,

se destacan:

- En el problema de la detección de armas y objetos que se manejan similarmente se pretende seguir reduciendo el número de falsos positivos y falsos negativos en las detecciones. Para ello se propone el uso de diferentes técnicas de binarización como *nested dichotomie* con la que se pretende ofrecer una alternativa a OVA y OVO.

  Con la técnica *nested dichotomie* se creará una agrupación de clases en estructura de árbol con la que se irá discriminando entre grupos de clases hasta llegar a la clase predicha reduciendo así los errores gracias al uso de modelos expertos por grupos.

- Para aumentar la robustez y capacidad de generalización del modelo de detección de infraestructuras críticas se propone aumentar el tamaño de la base de datos usando un método de preprocesado basado en las redes GAN (*Generative Adversarial Networks*) [RAGTRH20].

  Se entrenarán los generadores del modelo GAN para aprender las transformaciones inherentes de cada clase de infraestructura para más tarde generar imágenes nuevas a partir de las imágenes del entrenamiento. Esta propuesta es complementaria a las técnicas de *data augmentation* y *transfer learning*.

- Las series temporales de imágenes de satélite sufren de un desplazamiento aleatorio, es decir, el mismo arbusto puede aparecer desplazado varios píxeles hacia arriba, abajo, derecha y/o izquierda.

  Para paliar este problema en la detección de cambios de arbustos se plantea desarrollar un modelo de tracking para identificar correctamente cada arbusto en toda la serie temporal. La fase de identificación del modelo de tracking permite determinar el mismo arbusto en toda la serie temporal.

Bibliografía

[AB98] Allen C. D. and Breshears D. D. (1998) Drought-induced shift of a forest–woodland ecotone: rapid landscape response to climate variation. *Proceedings of the National Academy of Sciences* 95(25): 14839–14842.

[ABC+16] Abadi M., Barham P., Chen J., Chen Z., Davis A., Dean J., Devin M., Ghemawat S., Irving G., Isard M., *et al.* (2016) Tensorflow: a system for large-scale machine learning. In *OSDI*, volumen 16, pp. 265–283.

[Abe03] Abe S. (2003) Analysis of multiclass support vector machines. *Thyroid* 21(3): 3772.

[AMMR95] Anand R., Mehrotra K., Mohan C. K., and Ranka S. (1995) Efficient classification for multiclass problems using modular neural networks. *IEEE Transactions on Neural Networks* 6(1): 117–124.

[BBRL15] Byeon W., Breuel T. M., Raue F., and Liwicki M. (2015) Scene labeling with lstm recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3547–3555.

[BDZ11] Benedek C., Descombes X., and Zerubia J. (2011) Building development monitoring in multitemporal remotely sensed image pairs with stochastic birth-death dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(1): 33–50.

[BKC17] Badrinarayanan V., Kendall A., and Cipolla R. (2017) Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence* 39(12): 2481–2495.

[BKHB16] Bosch M., Kurtz Z., Hagstrom S., and Brown M. (2016) A multiple view stereo benchmark for satellite imagery. In *2016 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pp. 1–9. IEEE.

[BNLF21] Bjerreskov K. S., Nord-Larsen T., and Fensholt R. (2021) Classification of nemoral forests with fusion of multi-temporal sentinel-1 and 2 data. *Remote Sensing* 13(5): 950.

[BŞH18] Budak Ü., Şengür A., and Halici U. (2018) Deep convolutional neural networks for airport detection in remote sensing images. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4. IEEE.

[BVZ01] Boykov Y., Veksler O., and Zabih R. (2001) Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence* 23(11): 1222–1239.

[C+15] Chollet F. *et al.* (2015) Keras.

[CB91] Clark P. and Boswell R. (1991) Rule induction with cn2: Some recent improvements. In *European Working Session on Learning*, pp. 151–163. Springer.

[CC17] Chaurasia A. and Culurciello E. (2017) Linknet: Exploiting encoder representations for efficient semantic segmentation. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4. IEEE.

[CCZ+20] Cheng B., Collins M. D., Zhu Y., Liu T., Huang T. S., Adam H., and Chen L.-C. (2020) Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation.

[CFHL15] Chen X., Fang T., Huo H., and Li D. (2015) Measuring the effectiveness of various features for thematic information extraction from very high resolution remote sensing imagery. *IEEE Transactions on Geoscience and Remote Sensing* 53(9): 4837–4851.

[CFWM18] Christie G., Fendley N., Wilson J., and Mukherjee R. (2018) Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6172–6180.

[CGGGR19] Carranza-García M., García-Gutiérrez J., and Riquelme J. C. (2019) A framework for evaluating land use and land cover classification using convolutional neural networks. *Remote Sensing* 11(3): 274.

[CGHD19] Chen X., Girshick R., He K., and Dollár P. (2019) Tensormask: A foundation for dense object segmentation.

[CH16] Cheng G. and Han J. (2016) A survey on object detection in optical remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing* 117: 11–28.

[CHL17] Cheng G., Han J., and Lu X. (2017) Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE* 105(10): 1865–1883.

[CHZG14] Cheng G., Han J., Zhou P., and Guo L. (2014) Multi-class geospatial object detection and geographic image classification based on collection of part detectors. *ISPRS Journal of Photogrammetry and Remote Sensing* 98: 119–132.

[CJZ+17] Cai B., Jiang Z., Zhang H., Zhao D., and Yao Y. (2017) Airport detection using end-to-end convolutional neural network with hard example mining. *Remote Sensing* 9(11): 1198.

[CLW+21] Cheng G., Lang C., Wu M., Xie X., Yao X., and Han J. (2021) Feature enhancement network for object detection in optical remote sensing images. *Journal of Remote Sensing* 2021.

[CPK+14] Chen L.-C., Papandreou G., Kokkinos I., Murphy K., and Yuille A. L. (2014) Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062* .

[CPK+17] Chen L.-C., Papandreou G., Kokkinos I., Murphy K., and Yuille A. L. (2017) Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence* 40(4): 834–848.

[CSC18] Cruz R. M., Sabourin R., and Cavalcanti G. D. (2018) Dynamic classifier selection: Recent advances and perspectives. *Information Fusion* 41: 195–216.

[CTP+19] Castillo A., Tabik S., Pérez F., Olmos R., and Herrera F. (2019) Brightness guided preprocessing for automatic cold steel weapon detection in surveillance videos with deep learning. *Neurocomputing* 330: 151–161.

[CWL+20] Chen C., Wang B., Lu C. X., Trigoni N., and Markham A. (2020) A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence. *arXiv preprint arXiv:2006.12567* .

[CYW+16] Chen L.-C., Yang Y., Wang J., Xu W., and Yuille A. L. (2016) Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3640–3649.

[CYY+18] Cheng G., Yang C., Yao X., Guo L., and Han J. (2018) When deep learning meets metric learning: Remote sensing image scene classification via learning discriminative cnns. *IEEE transactions on geoscience and remote sensing* 56(5): 2811–2821.

[CZH16] Cheng G., Zhou P., and Han J. (2016) Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing* 54(12): 7405–7415.

[DDS+09] Deng J., Dong W., Socher R., Li L.-J., Li K., and Fei-Fei L. (2009) Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255. Ieee.

[DJS+18] Ding H., Jiang X., Shuai B., Liu A. Q., and Wang G. (2018) Context contrasted feature and gated multi-scale aggregation for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2393–2402.

[DMC15] Dhanachandra N., Manglem K., and Chanu Y. J. (2015) Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science* 54: 764–771.

[DP14] Deng L. and Platt J. (2014) Ensemble deep learning for speech recognition. In *Proc. Interspeech*.

[DRF16] Dai K. J. and R-FCN Y. L. (2016) Object detection via region-based fully convolutional networks. arxiv preprint. *arXiv preprint arXiv:1605.06409* .

[FCB+09] Fernández A., Calderón M., Barrenechea E., Bustince H., and Herrera F. (2009) Enhancing fuzzy rule based systems in multi-classification using pairwise coupling with preference relations. *EUROFUSE* 9: 39–46.

[FGG+18] Fernández A., García S., Galar M., Prati R. C., Krawczyk B., and Herrera F. (2018) *Learning from Imbalanced Data Sets*. Springer.

[FLW+19] Fu J., Liu J., Wang Y., Zhou J., Wang C., and Lu H. (2019) Stacked deconvolutional network for semantic segmentation. *IEEE Transactions on Image Processing* .

[FP11] Forsyth D. and Ponce J. (2011) *Computer vision: A modern approach.* Prentice hall.

[Fri96] Friedman J. H. (1996) Another approach to polychotomous classification. *Technical Report, Statistics Department, Stanford University* .

[FSB19] Fu C.-Y., Shvets M., and Berg A. C. (2019) Retinamask: Learning to predict masks improves state-of-the-art single-shot detection for free.

[FWC19] Flood N., Watson F., and Collett L. (2019) Using a u-net convolutional neural network to map woody vegetation extent from high resolution satellite imagery across queensland, australia. *International Journal of Applied Earth Observation and Geoinformation* 82: 101897.

[GASC+20] Guirado E., Alcaraz-Segura D., Cabello J., Puertas-Ruíz S., Herrera F., and Tabik S. (2020) Tree cover estimation in global drylands from space using deep learning. *Remote Sensing* 12(3): 343.

[GC17] Guidici D. and Clark M. L. (2017) One-dimensional convolutional neural network land-cover classification of multi-seasonal hyperspectral imagery in the san francisco bay area, california. *Remote Sensing* 9(6): 629.

[GFB+11] Galar M., Fernández A., Barrenechea E., Bustince H., and Herrera F. (2011) An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition* 44(8): 1761–1776.

[GFBH15] Galar M., Fernández A., Barrenechea E., and Herrera F. (2015) Drcw-ovo: distance-based relative competence weighting combination for one-vs-one strategy in multi-class problems. *Pattern recognition* 48(1): 28–42.

[GRTL+19] Gómez-Ríos A., Tabik S., Luengo J., Shihavuddin A., Krawczyk B., and Herrera F. (2019) Towards highly accurate coral texture images classification using deep convolutional neural networks and data augmentation. *Expert Systems with Applications* 118: 315–328.

[GTR+19] Guirado E., Tabik S., Rivas M. L., Alcaraz-Segura D., and Herrera F. (2019) Whale counting in satellite and aerial images with deep learning. *Scientific reports* 9(1): 1–12.

[GZ03] García D. and Zamora R. (2003) Persistence, multiple demographic strategies and conservation in long-lived mediterranean plants. *Journal of Vegetation Science* 14(6): 921–926.

[HB08] Hüllermeier E. and Brinker K. (2008) Learning valued preference structures for solving classification problems. *Fuzzy Sets and Systems* 159(18): 2337–2352.

[HBDS15] Hosang J., Benenson R., Dollár P., and Schiele B. (2015) What makes for effective detection proposals? *IEEE transactions on pattern analysis and machine intelligence* 38(4): 814–830.

[HDQ19] He J., Deng Z., and Qiao Y. (2019) Dynamic multi-scale filters for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3562–3572.

[HDZ+19] He J., Deng Z., Zhou L., Wang Y., and Qiao Y. (2019) Adaptive pyramid context network for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7519–7528.

[HGDG17] He K., Gkioxari G., Dollár P., and Girshick R. (2017) Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.

[HGDG18] He K., Gkioxari G., Dollár P., and Girshick R. (2018) Mask r-cnn.

[HH09] Huhn J. C. and Hullermeier E. (2009) Fr3: A fuzzy rule learner for inducing reliable classifiers. *IEEE Transactions on Fuzzy Systems* 17(1): 138.

[HHS+20] Hatamizadeh A., Hoogi A., Sengupta D., Lu W., Wilcox B., Rubin D., and Terzopoulos D. (2020) Deep active lesion segmentation.

[HP05] Hampe A. and Petit R. J. (2005) Conserving biodiversity under climate change: the rear edge matters. *Ecology letters* 8(5): 461–467.

[HRS+17] Huang J., Rathod V., Sun C., Zhu M., Korattikara A., Fathi A., Fischer I., Wojna Z., Song Y., Guadarrama S., *et al.* (2017) Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7310–7311.

[HSS17] Hu J., Shen L., and Sun G. (2017) Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507* 7.

[HSYF21] He Q., Sun X., Yan Z., and Fu K. (2021) Dabnet: Deformable contextual and boundary-weighted network for cloud detection in remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing* .

[HT98] Hastie T. and Tibshirani R. (1998) Classification by pairwise coupling. In *Advances in neural information processing systems*, pp. 507–513.

[HTL+18] Hung W.-C., Tsai Y.-H., Liou Y.-T., Lin Y.-Y., and Yang M.-H. (2018) Adversarial learning for semi-supervised semantic segmentation. *arXiv preprint arXiv:1802.07934* .

[HXW+17] Huang Q., Xia C., Wu C., Li S., Wang Y., Song Y., and Kuo C.-C. J. (2017) Semantic segmentation with reverse attention. *arXiv preprint arXiv:1707.06426* .

[HZ14] Herrero A. and Zamora R. (2014) Plant responses to extreme climatic events: a field test of resilience capacity at the southern range edge. *Plos one* 9(1): e87842.

[HZRS16] He K., Zhang X., Ren S., and Sun J. (2016) Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

[JHP+19] Jiang X., Hadid A., Pang Y., Granger E., and Feng X. (2019) *Deep Learning in Object Detection and Recognition*. Springer.

[JMP09] Jump A. S., Mátyás C., and Peñuelas J. (2009) The altitude-for-latitude disparity in the range retractions of woody species. *Trends in ecology & evolution* 24(12): 694–701.

[KGHD19] Kirillov A., Girshick R., He K., and Dollár P. (2019) Panoptic feature pyramid networks.

[KPD90] Knerr S., Personnaz L., and Dreyfus G. (1990) Single-layer learning revisited: a stepwise procedure for building and training a neural network. In *Neurocomputing*, pp. 41–50. Springer.

[KSH12] Krizhevsky A., Sutskever I., and Hinton G. E. (2012) Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105.

[Kul02] Kullman L. (2002) Rapid recent range-margin rise of tree and shrub species in the swedish scandes. *Journal of ecology* 90(1): 68–77.

[KWT88] Kass M., Witkin A., and Terzopoulos D. (1988) Snakes: Active contour models. *International journal of computer vision* 1(4): 321–331.

[LAE+16] Liu W., Anguelov D., Erhan D., Szegedy C., Reed S., Fu C.-Y., and Berg A. C. (2016) Ssd: Single shot multibox detector. In *European conference on computer vision*, pp. 21–37. Springer.

[LCCV16] Luc P., Couprie C., Chintala S., and Verbeek J. (2016) Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408* .

[LCW+18] Liu Y., Chen X., Wang Z., Wang Z. J., Ward R. K., and Wang X. (2018) Deep learning for pixel-level image fusion: Recent advances and future prospects. *Information Fusion* 42: 158–173.

[LCZ+19] Li Y., Chen X., Zhu Z., Xie L., Huang G., Du D., and Wang X. (2019) Attention-guided unified network for panoptic segmentation.

[LDG+17] Lin T.-Y., Dollár P., Girshick R., He K., Hariharan B., and Belongie S. (2017) Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125.

[LG05] Lei H. and Govindaraju V. (2005) Half-against-half multi-class support vector machines. In *International Workshop on Multiple Classifier Systems*, pp. 156–164. Springer.

[LGGRG+20] Luengo J., García-Gil D., Ramírez-Gallego S., García S., and Herrera F. (2020) Big data preprocessing. *Cham: Springer* .

[LGM+08] Lenoir J., Gégout J.-C., Marquet P., De Ruffray P., and Brisse H. (2008) A significant upward shift in plant species optimum elevation during the 20th century. *science* .

[LJS+08] Lee G., Jordan E., Shishko R., de Weck O., Armar N., and Siddiqi A. (2008) Spacenet: Modeling and simulating space logistics. In *AIAA SPACE 2008 Conference & Exposition*, page 7747.

[LKM+18] Lam D., Kuzma R., McGee K., Dooley S., Laielli M., Klaric M., Bulatov Y., and McCord B. (2018) xview: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856* .

[LLL+15] Liu Z., Li X., Luo P., Loy C.-C., and Tang X. (2015) Semantic image segmentation via deep parsing network. In *Proceedings of the IEEE international conference on computer vision*, pp. 1377–1385.

[LLS14] Li Z., Liu Z., and Shi W. (2014) Semiautomatic airport runway extraction using a line-finder-aided level set evolution. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7(12): 4738–4749.

[LLST20] Lazarow J., Lee K., Shi K., and Tu Z. (2020) Learning instance occlusion for panoptic segmentation.

[LLY+21] Li Z., Liu F., Yang W., Peng S., and Zhou J. (2021) A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems* .

[LM15] Liu K. and Mattyus G. (2015) Fast multiclass vehicle detection on aerial images. *IEEE Geoscience and Remote Sensing Letters* 12(9): 1938–1942.

[LMB+14] Lin T.-Y., Maire M., Belongie S., Hays J., Perona P., Ramanan D., Dollár P., and Zitnick C. L. (2014) Microsoft COCO: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer.

[LOB11] Laguna J. O., Olaya A. G., and Borrajo D. (2011) A dynamic sliding window approach for activity recognition. In *International conference on user modeling, adaptation, and personalization*, pp. 219–230. Springer.

[LP20] Lee Y. and Park J. (2020) Centermask : Real-time anchor-free instance segmentation.

[LQLY19] Liu S., Qi Z., Li X., and Yeh A. G.-O. (2019) Integration of convolutional neural networks and object-based post-classification refinement for land use and land cover mapping with optical and sar data. *Remote Sensing* 11(6): 690.

[LQQ+18] Liu S., Qi L., Qin H., Shi J., and Jia J. (2018) Path aggregation network for instance segmentation.

[LRFS03] LLOYD A. H., RUPP T. S., FASTIE C. L., and STARFIELD A. M. (2003) Patterns and dynamics of treeline advance on the seward peninsula, alaska. *Journal of geophysical research* 108(D2): ALT2–1.

[LSD15] Long J., Shelhamer E., and Darrell T. (2015) Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

[LSF+16] Liang X., Shen X., Feng J., Lin L., and Yan S. (2016) Semantic object parsing with graph lstm. In *European Conference on Computer Vision*, pp. 125–143. Springer.

[LSVDHR16] Lin G., Shen C., Van Den Hengel A., and Reid I. (2016) Efficient piecewise training of deep structured models for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3194–3203.

[LW07] Lu D. and Weng Q. (2007) A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing* 28(5): 823–870.

[LWC+20] Li K., Wan G., Cheng G., Meng L., and Han J. (2020) Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS Journal of Photogrammetry and Remote Sensing* 159: 296–307.

[LXAW18] Li H., Xiong P., An J., and Wang L. (2018) Pyramid attention network for semantic segmentation. *arXiv preprint arXiv:1805.10180* .

[LXZ+19] Li S., Xu Y., Zhu M., Ma S., and Tang H. (2019) Remote sensing airport detection based on end-to-end deep transferable convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters* 16(10): 1640–1644.

[LZW+19] Li X., Zhong Z., Wu J., Yang Y., Lin Z., and Liu H. (2019) Expectation-maximization attention networks for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9167–9176.

[LZYF19] Lu X., Zhang Y., Yuan Y., and Feng Y. (2019) Gated and axis-concentrated localization network for remote sensing object detection. *IEEE Transactions on Geoscience and Remote Sensing* 58(1): 179–192.

[MBP+20] Minaee S., Boykov Y., Porikli F., Plaza A., Kehtarnavaz N., and Terzopoulos D. (2020) Image segmentation using deep learning: A survey.

[MBP+21] Minaee S., Boykov Y. Y., Porikli F., Plaza A. J., Kehtarnavaz N., and Terzopoulos D. (2021) Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* .

[MJ15] Matías L. and Jump A. S. (2015) Asymmetric changes of growth and reproductive investment herald altitudinal and latitudinal range shifts of two woody species. *Global Change Biology* 21(2): 882–896.

[MM13] Moranduzzo T. and Melgani F. (2013) Automatic car counting method for unmanned aerial vehicle images. *IEEE Transactions on Geoscience and Remote Sensing* 52(3): 1635–1647.

[MTK+18] Marcos D., Tuia D., Kellenberger B., Zhang L., Bai M., Liao R., and Urtasun R. (2018) Learning deep structured active contours end-to-end.

[MV21] Mohan R. and Valada A. (2021) Efficientps: Efficient panoptic segmentation.

[NHH15] Noh H., Hong S., and Han B. (2015) Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528.

[NN04] Nock R. and Nielsen F. (2004) Statistical region merging. *IEEE Transactions on pattern analysis and machine intelligence* 26(11): 1452–1458.

[OHA+18] Oshri B., Hu A., Adelson P., Chen X., Dupas P., Weinstein J., Burke M., Lobell D., and Ermon S. (2018) Infrastructure quality assessment in africa using satellite imagery and deep learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 616–625.

[Orl78] Orlovsky S. (1978) Decision-making with a fuzzy preference relation. *Fuzzy sets and systems* 1(3): 155–167.

[OTC+19] Olmos R., Tabik S., Castillo A., Pérez F., and Herrera F. (2019) A binocular image fusion approach for minimizing false positives in handgun detection with deep learning. *Information Fusion* 49: 271 – 280.

[OTH18] Olmos R., Tabik S., and Herrera F. (2018) Automatic handgun detection alarm in videos using deep learning. *Neurocomputing* 275: 66–72.

[ÖY17] Öztürk K. and Yilmaz M. B. (2017) A comparison of classification approaches for deep face recognition. In *Computer Science and Engineering (UBMK), 2017 International Conference on*, pp. 227–232. IEEE.

[Pau94] Pauli G. G. M. G. H. (1994) Climate effects on mountain plants. *Nature* 369: 448.

[PB03] Penuelas J. and Boada M. (2003) A global change-induced biome shift in the montseny mountains (ne spain). *Global change biology* 9(2): 131–140.

[PBCK19] Porzi L., Bulò S. R., Colovic A., and Kontschieder P. (2019) Seamless scene segmentation.

[PCD15] Pinheiro P. O., Collobert R., and Dollar P. (2015) Learning to segment object candidates.

[PCKC16] Paszke A., Chaurasia A., Kim S., and Culurciello E. (2016) Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147* .

[PCST00] Platt J. C., Cristianini N., and Shawe-Taylor J. (2000) Large margin dags for multiclass classification. In *Advances in neural information processing systems*, pp. 547–553.

[PGD+12] Pauli H., Gottfried M., Dullinger S., Abdaladze O., Akhalkatsi M., Alonso J. L. B., Coldea G., Dick J., Erschbamer B., Calzado R. F., *et al.* (2012) Recent plant diversity changes on europe's mountain summits. *Science* 336(6079): 353–355.

[PLCD16] Pinheiro P. O., Lin T.-Y., Collobert R., and Dollàr P. (2016) Learning to refine object segments.

[PPR18] Pathak A. R., Pandey M., and Rautaray S. (2018) Application of deep learning for object detection. *Procedia computer science* 132: 1706–1717.

[PW10] Pele O. and Werman M. (2010) The quadratic-chi histogram distance family. In *European conference on computer vision*, pp. 749–762. Springer.

[PY10] Pan S. J. and Yang Q. (2010) A survey on transfer learning. ieee transactions on knowledge and data engineering. *22 (10): 1345* 1359.

[RAGTRH20] Rey-Area M., Guirado E., Tabik S., and Ruiz-Hidalgo J. (2020) Fucitnet: Improving the generalization of deep learning networks by the fusion of learned class-inherent transformations. *Information Fusion* 63: 188–195.

[RDGF16] Redmon J., Divvala S., Girshick R., and Farhadi A. (2016) You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

[RDS+15] Russakovsky O., Deng J., Su H., Krause J., Satheesh S., Ma S., Huang Z., Karpathy A., Khosla A., Bernstein M., *et al.* (2015) Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115(3): 211–252.

[RFB15] Ronneberger O., Fischer P., and Brox T. (2015) U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer.

[RG14] Rocha A. and Goldenstein S. K. (2014) Multiclass from binary: Expanding one-versus-all, one-versus-one and ecoc-based approaches. *IEEE Transactions on Neural Networks and Learning Systems* 25(2): 289–302.

[RHBN16] Rupprecht C., Huaroc E., Baust M., and Navab N. (2016) Deep active contours.

[RHGS15a] Ren S., He K., Girshick R., and Sun J. (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99.

[RHGS15b] Ren S., He K., Girshick R., and Sun J. (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. In Cortes C., Lawrence N., Lee D., Sugiyama M., and Garnett R. (Eds.) *Advances in Neural Information Processing Systems*, volumen 28. Curran Associates, Inc.

[Ros76] Rosenfeld A. (1976) *Digital picture processing*. Academic press.

[SBK19] Sofiiuk K., Barinova O., and Konushin A. (2019) Adaptis: Adaptive instance selection network.

[SEDGS03] SANZ-ELORZA M., Dana E. D., González A., and Sobrino E. (2003) Changes in the high-mountain vegetation of the central iberian peninsula as a probable sign of global warming. *Annals of Botany* 92(2): 273–280.

[SGM+21] Safonova A., Guirado E., Maglinets Y., Alcaraz-Segura D., and Tabik S. (2021) Olive tree biovolume from uav multi-resolution image segmentation with mask r-cnn. *Sensors* 21(5): 1617.

[SIVA16] Szegedy C., Ioffe S., Vanhoucke V., and Alemi A. (2016) Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261* .

[SIVA17] Szegedy C., Ioffe S., Vanhoucke V., and Alemi A. A. (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

[SK19] Shorten C. and Khoshgoftaar T. M. (2019) A survey on image data augmentation for deep learning. *Journal of Big Data* 6(1): 1–48.

[SLJ+15] Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., and Rabinovich A. (2015) Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

[SSB17] Sommer L. W., Schuchert T., and Beyerer J. (2017) Deep learning based multi-category object detection in aerial images. In *Automatic Target Recognition XXVII*, volumen 10202, page 1020209. International Society for Optics and Photonics.

[STE13] Szegedy C., Toshev A., and Erhan D. (2013) Deep neural networks for object detection. In Burges C. J. C., Bottou L., Welling M., Ghahramani Z., and Weinberger K. Q. (Eds.) *Advances in Neural Information Processing Systems*, volumen 26. Curran Associates, Inc.

[SVI+16] Szegedy C., Vanhoucke V., Ioffe S., Shlens J., and Wojna Z. (2016) Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.

[SWW+21] Sun X., Wang B., Wang Z., Li H., Li H., and Fu K. (2021) Research progress on few-shot learning for remote sensing image interpretation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14: 2387–2402.

[SZ14] Simonyan K. and Zisserman A. (2014) Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* .

[Sze10] Szeliski R. (2010) *Computer vision: algorithms and applications*. Springer Science & Business Media.

[TN18] Taylor L. and Nitschke G. (2018) Improving deep learning with generic data augmentation. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1542–1547. IEEE.

[TPHPH17] Tabik S., Peralta D., Herrera-Poyatos A., and Herrera F. (2017) A snapshot of image pre-processing for convolutional neural networks: case study of mnist. *International Journal of Computational Intelligence Systems* 10(1): 555–568.

[TZD+17] Tang T., Zhou S., Deng Z., Zou H., and Lei L. (2017) Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining. *Sensors* 17(2): 336.

[VCR+16] Visin F., Ciccone M., Romero A., Kastner K., Cho K., Bengio Y., Matteucci M., and Courville A. (2016) Reseg: A recurrent neural network-based model for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 41–48.

[VELB18] Van Etten A., Lindenbaum D., and Bacastow T. M. (2018) Spacenet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232* .

[WCY+18] Wang P., Chen P., Yuan Y., Liu D., Huang Z., Hou X., and Cottrell G. (2018) Understanding convolution for semantic segmentation. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pp. 1451–1460. IEEE.

[WCZ16] Wang D., Cui P., and Zhu W. (2016) Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1225–1234.

[WLB+19] Weir N., Lindenbaum D., Bastidas A., Etten A. V., McPherson S., Shermeyer J., Kumar V., and Tang H. (2019) Spacenet mvoi: a multi-view overhead imagery dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 992–1001.

[WLW04] Wu T.-F., Lin C.-J., and Weng R. C. (2004) Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* 5(Aug): 975–1005.

[WXX+14] Wu W., Xia R., Xiang W., Hui B., Chang Z., Liu Y., and Zhang Y. (2014) Recognition of airport runways in flir images based on knowledge. *IEEE Geoscience and Remote Sensing Letters* 11(9): 1534–1538.

[XBD+18] Xia G.-S., Bai X., Ding J., Zhu Z., Belongie S., Luo J., Datcu M., Pelillo M., and Zhang L. (2018) Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3974–3983.

[XF17] Xiang Y. and Fox D. (2017) Da-rnn: Semantic mapping with data associated recurrent neural networks. *arXiv preprint arXiv:1703.03098* .

[XGL+17] Xiao Z., Gong Y., Long Y., Li D., Wang X., and Liu H. (2017) Airport detection based on a multiscale fusion feature for optical remote sensing images. *IEEE Geoscience and Remote Sensing Letters* 14(9): 1469–1473.

[XK17] Xia X. and Kulis B. (2017) W-net: A deep model for fully unsupervised image segmentation. *arXiv preprint arXiv:1711.08506* .

[XLZ+19] Xiong Y., Liao R., Zhao H., Hu R., Bai M., Yumer E., and Urtasun R. (2019) Upsnet: A unified panoptic segmentation network.

[XSS+20] Xie E., Sun P., Song X., Wang W., Liang D., Shen C., and Luo P. (2020) Polarmask: Single shot instance segmentation with polar representation.

[XXZ18] Xiang T.-Z., Xia G.-S., and Zhang L. (2018) Mini-uav-based remote sensing: techniques, applications and prospectives. *preprint* .

[XZL+18] Xu Y., Zhu M., Li S., Feng H., Ma S., and Che J. (2018) End-to-end airport detection in remote sensing images combining cascade region proposal networks and multi-threshold detection networks. *Remote Sensing* 10(10): 1516.

[YCW20] Yuan Y., Chen X., and Wang J. (2020) Object-contextual representations for semantic segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pp. 173–190. Springer.

[YGK17] Yu M., Gong L., and Kollias S. (2017) Computer vision based fall detection by a convolutional neural network. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 416–420. ACM.

[YHG+18] Yuan Y., Huang L., Guo J., Zhang C., Chen X., and Wang J. (2018) Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916* .

[YHX+19] Yuan J., Hou X., Xiao Y., Cao D., Guan W., and Nie L. (2019) Multi-criteria active deep learning for image classification. *Knowledge-Based Systems* 172: 86–94.

[YN10] Yang Y. and Newsam S. (2010) Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pp. 270–279.

[YWP+18] Yu C., Wang J., Peng C., Gao C., Yu G., and Sang N. (2018) Learning a discriminative feature network for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1857–1866.

[YYZ+18] Yang M., Yu K., Zhang C., Li Z., and Yang K. (2018) Denseaspp for semantic segmentation in street scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3684–3692.

[ZBX+19] Zhang C., Bi J., Xu S., Ramentol E., Fan G., Qiao B., and Fujita H. (2019) Multi-imbalance: An open-source software for multi-class imbalance learning. *Knowledge-Based Systems* .

[ZNDX17] Zhang P., Niu X., Dou Y., and Xia F. (2017) Airport detection on optical satellite images using deep convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters* 14(8): 1183–1187.

[ZSP+19] Zhang C., Sargent I., Pan X., Li H., Gardiner A., Hare J., and Atkinson P. M. (2019) Joint deep learning for land cover and land use classification. *Remote sensing of environment* 221: 173–187.

[ZSQ+17] Zhao H., Shi J., Qi X., Wang X., and Jia J. (2017) Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890.

[ZSS+21] Zheng L., Shi P., Song M., Zhou T., Zong N., and Zhang X. (2021) Climate sensitivity of high altitude tree growth across the hindu kush himalaya. *Forest Ecology and Management* 486: 118963.

[ZWZ15] Zhu D., Wang B., and Zhang L. (2015) Airport target detection in remote sensing images: A new method based on two-way saliency. *IEEE Geoscience and Remote Sensing Letters* 12(5): 1096–1100.

[ZYCL18] Zhang Q., Yang L. T., Chen Z., and Li P. (2018) A survey on deep learning for big data. *Information Fusion* 42: 146–157.

[ZYFL19] Zhang Y., Yuan Y., Feng Y., and Lu X. (2019) Hierarchical and robust convolutional neural network for very high-resolution remote sensing object detection. *IEEE Transactions on Geoscience and Remote Sensing* 57(8): 5535–5548.